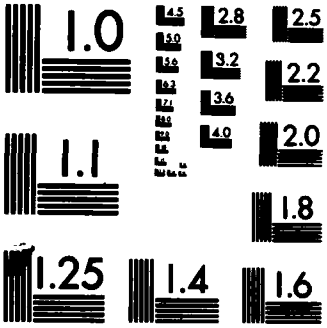
AD-A138 425 ROBUST FLIGHT CONTROLLERS(U) AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOOL OF ENGINEERRNG J M HOWEY DEC 83 AFIT/GAE/EE/83D-2 1/3

UNCLASSIFIED F/G 12/1 NL


1.0
4.5
5.0
5.6
6.3
7.1
8.0
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

ROBUST FLIGHT CONTROLLERS

THESIS

AFIT/GAE/EE/83D-2

Jean M. Howey
2Lt USAF

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GAE/EE/83D-2

ROBUST FLIGHT CONTROLLERS

THESIS

AFIT/GAE/EE/83D-2

Jean M. Howey
2Lt USAF

Approved for public release; distribution unlimited

AFIT/GAE/EE/83D-2

ROBUST FLIGHT CONTROLLERS

THESIS

Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology
Air University
in Partial Fulfillment of the
Requirements for the Degree of
Master of Science

by
Jean M. Howey, B.S.
2Lt USAF
Graduate Aeronautical Engineering

December 1983

Approved for public release; distribution unlimited

Preface

In the past two decades, one of the major stumbling blocks in applying optimal control theory to flight control problems has been that measurements of all the states of the system are not available. Thus, it is necessary to use some type of observer or filter to reconstruct the states of the aircraft. However, when this is done, all guarantees of desirable stability robustness properties are lost. This thesis addresses this problem and evaluates the success of some techniques to recover the good stability robustness properties associated with full-state feedback.

I wish to thank my advisor, Professor Peter S. Maybeck for his invaluable guidance during the course of this research and for his thorough editing of this document. In addition, I would like to thank my committee members, Professors John J. D'Azzo and Robert Calico for their suggestions and comments. Finally, I wish to thank Captain Richard Floyd for his assistance in modifying and correcting the computer programs used to accomplish this project.

Contents

Page

Contents (cont'd)

Contents (cont'd)

List of Figures

List of Figures (Cont'd)

List of Figures (Cont'd)

List of Figures (Cont'd)

List of Figures (Cont'd)

List of Figures (Cont'd)

List of Figures (Cont'd)

List of Tables

Abstract

This study examines the concept of robustifying a controlled system against differences which may exist between the real world system and the low-order design model upon which the controller design is based. The types of controllers considered are based upon the Linear system model, Quadratic cost, and Gaussian (LQG) noise process methodology of optimal control theory. It is assumed that full-state feedback is not available and a Kalman filter is employed to provide state estimates to the controller. Both continuous-time and sampled-data controllers are considered.

Two robustification techniques are considered. The first is the method of injecting zero-mean white Gaussian noise into the design model at the point of entry of the control inputs during the process of tuning the Kalman filter. The second method is an extension of the first, where the white noise is replaced by a time-correlated noise. This allows the primary strength of the noise to be concentrated only in the frequency range where robustification is desired. Comparing the results of applying the two methods allows a designer to make a trade-off between the amount of desired robustification and the performance degradation at the design conditions which occurs when the techniques are applied.

Both methods are found to improve substantially the robustness properties of the controllers considered. For the specific flight control problem considered in this thesis, the technique of injecting white input noise into the design model produced the desired degree of robustification without prohibitively degrading performance at the design conditions.

The second method, though effective, did not yield substantial enough performance benefits over the first to warrant use in actual implementation.

ROBUST FLIGHT CONTROLLERS

I. Introduction

1.1 Background

A vital element of a flight control system is that it be "robust". Robustness implies that the controller provides adequate (i.e., stable closed-loop) performance over a wide range of operating conditions and system parameters. For example, finite-dimension models of the dynamics of an aircraft are typically generated by linearizing the aircraft equations of motion about a limited number of specific equilibrium flight conditions. Controllers are then designed for these "trim" conditions. However, at flight conditions other than the specific trim condition used for controller design, the closed-loop performance of the controller system may be inadequate or unstable. That is, at the off-design flight condition, the aircraft parameters (upon which the design is based) have changed sufficiently to make performance of the controller inadequate. One method used in the past has been to schedule feedback gains for the controller so as to compensate for the change in system parameters. However, it is desirable to have enlarged regions about nominal design conditions within which a specific controller design will yield adequate performance. This would allow linear perturbation techniques to be used with more confidence and may also reduce the number of required design conditions about which such perturbations are defined. It may even be desirable to have a fixed controller which is sufficiently robust to produce acceptable performance for all conditions in the aircraft's flight envelope.

Robustness is also a concern when controllers are designed using purposefully reduced order models. A robust controller will provide stable closed-loop performance even when states of the real-world system have been ignored in the design model. A third robustness issue is survivability. Robustness can also imply the ability to maintain closed-loop stability if, for instance, part of the flight control system is lost due to ground fire: i.e., the "true" system is vastly different than the system model upon which the controller was designed. Such a robust controller will provide a stable, though degraded, aircraft performance while adaptation algorithms attempt to discern what system elements have been lost and to determine an appropriate modification of controller characteristics for future use.

In the 1960's, modern control theory methods, such as optimal control theory, showed promise in application to flight control problems. One drawback of applying optimal control theory techniques to flight control problems, however, is that the resulting controllers require full-state feedback, but measurements of all states are generally not available. Thus, it becomes necessary to include a filter or observer in the controlled system to estimate the states. However, once an observer has been inserted into the loop, stability robustness becomes a major concern (Ref 5).

Recently, efforts to improve the robustness properties of observer-based controllers have included a technique, developed by J.C. Doyle and G. Stein (Ref 6), which injects white noise into a controller system model at the point of entry of the control inputs during the process of tuning the Kalman filter. It is claimed that, as the strength of the input noise is increased, the filter-based controller will asymptotically recover the good robustness properties of a full-state feedback system in the continuous-time case. A disadvantage of this technique is that the additional noise can degrade the performance of the system at the design conditions.

A natural extension to the idea of injecting white input noise into a system model is to consider time-correlated noise. This allows the robustification technique to be applied only over a desired frequency range rather than over all frequencies as in the original Doyle and Stein method. Thus, the degradation in performance due to the additional noise can be reduced (Ref 15;16;28).

The techniques described above are applied in this thesis to a specific aircraft flight control problem. Two types of controllers are considered. The first is an optimal Linear Quadratic Gaussian (LQG) regulator. Available software allows the design of a continuous-time and a digital controller. The second type of controller is an optimal LQG-based Proportional-plus-Integral digital controller. For both types of controller, a Kalman filter is implemented to provide estimates of the states of the system. The success of applying the robustification techniques is determined by designing a Kalman filter and controller for one trim condition, then evaluating the performance of the controller at an off-design flight condition, as well as designing the filter and controller on the basis of purposely reduced-order models for computational loading reasons. Time histories of the mean and standard deviations of the aircraft states and controls are examined. Additionally, it is desired to compare the performance of the controller at the design conditions with no input noise, white noise and time-correlated noise used for filter retuning (robustification) to determine how much the performance is degraded by the input of the noise, and also to determine if performance benefits can be gained by using time-correlated noise as opposed to white noise.

1.2 Problem

The primary objectives of this thesis are:

1. To apply the robustification techniques of injecting white or time-correlated noise into a system model at the control entry points during filter tuning to a flight control problem for a high performance aircraft.
2. To extend the techniques for LQG regulators to LQG-based PI controllers.
3. To evaluate the robustness properties of the controller designs by performing covariance analyses for the controlled systems at both the design flight condition and other off-design conditions within the aircraft's operational flight envelope, using a purposefully reduced-order design model. Robustified designs are to be compared to unrobustified designs and also to full-state feedback designs when possible.

1.3 Sequence of Presentation

The body of this thesis is contained in Chapter II-V. Chapter II presents the equations used for designing and evaluating optimal deterministic LQG regulators for both the continous- and discrete-time case.

Also included is the application of the Doyle and Stein technique to continuous-time systems and several extensions of the technique to discrete-time systems.

Chapter III presents the equations necessary to design and evaluate optimal LQG-based PI controllers for the discrete-time case. Then the Doyle and Stein technique is applied to systems employing PI controllers. As pointed out in the text, robustification by this technique is a procedure which affects only the Kalman filter design; thus, the same method is used for PI controllers as for regulators in this thesis.

Chapter IV examines the idea of injecting time-correlated noise rather than white noise into a controlled system model during tuning of the filter, to improve the tradeoff of the controller's robustness properties versus performance degradation at design conditions. First, a stochastic process model (shaping filter) is developed for the noise process which is then augmented with the system state differential equations. Then, the types of time-correlated noise of interest to this thesis are considered, including a discussion of the specific shaping filters to generate the desired noise process.

Chapter V presents the model of a high-performance aircraft to be used in this thesis. A design flight condition is chosen and linearized perturbation equations of motion are developed for the aircraft about that operating point. The design model for the Kalman filter and controller is purposefully reduced in order from the full set of linearized equations so that this aspect of robustness can be examined. Finally, models at other flight conditions are listed with which the robustness of the system to parameter changes will be evaluated.

The findings of this thesis, results and conclusions are presented in Chapter VI. Recommendations for further research are made in Chapter VII.

Four appendices are included. The first presents a generic format for a controller into which the types of controllers in Chapters II and III can be rearranged. The usefulness of the format becomes apparent in a performance analysis, when the same set of equations can be used to evaluate the performance of any controller in the standard format.

Appendix B lists the source code for a Fortran program used in designing LQG regulators. Included are a discussion of how the program is executed and modifications from a previous version to allow the input of time-correlated noise (Ref 21).

Appendix C lists the modification to the software of Reference 13 and 30 to allow the input of white and time-correlated noise into the system model. The software is an interactive program used for designing PI regulators with a Command Generator Tracker in the feed-forward loop. This thesis will exploit only the PI regulator design capabilities.

Appendix D includes further performance analysis results in addition to the findings in Chapter VI. It was stated in Reference 6 that the Doyle and Stein technique is not guaranteed to improve robustness if the design model is non-minimum phase (i.e., there are transmission zeroes in the right-half s-plane). Appendix D shows a case where the design model was non-minimum phase and where the addition of input noise to the system model has actually drive an initially stable closed-loop system to be unstable.

II. LQG REGULATORS

2.1 Introduction

This chapter presents the equations used in designing optimal Linear Quadratic Gaussian (LQG) controllers for a system modelled as linear, time-invariant, and driven by zero-mean, white Gaussian noise and deterministic inputs, subject to quadratic costs for defining optimality criteria. The model used for this thesis is described in Chapter V. The methods used are taken primarily from Reference 24 unless otherwise stated.

In the first section, the controller equations for a continuous-time system having continuous-time measurements are given. The structures of controllers assuming perfect access to all the states of a system versus controllers with a Kalman filter to estimate states are examined. Next, the equations needed to evaluate the performance of controller designs are given. Finally, a technique developed by Doyle and Stein (Ref 6) to robustify the Kalman filter for continuous-time systems is presented.

The fourth section contains the controller equations for a continuous-time system having sampled-data measurements. Next the performance analysis equations for the sampled-data system are developed.

The final sections contain alternative methods for applying the Doyle and Stein technique to sampled-data systems. Three possible basic approaches are simply to discretize the controller designed for the continuous system or to modify the technique in one of two ways so that it applies directly to a discrete system.

All methods described in this chapter are incorporated in a Fortran program originally written by Captain Eric Lloyd and modified to some extent by this author. For source codes and information about the program, see Reference 21. Modifications and additions to the program are listed in Appendix B.

2.2 Continuous-Time Controllers

The state description for an important class of continuous-time systems is given by the linear stochastic differential equation

$$\dot{\underline{x}}(t) = F(t)\,\underline{x}(t) + B(t)\,\underline{u}(t) + G(t)\,\underline{w}(t) \tag{2-1}$$

where $\underline{w}(t)$ is a white Gaussian noise with statistics

$$E\{\underline{w}(t)\} = \underline{0} \tag{2-2a}$$

$$E\{\underline{w}(t)\,\underline{w}^T(t+\tau)\} = Q(t)\delta(\tau) \tag{2-2b}$$

An optimal LQG controller for the system minimizes the cost functional

$$J_c = E\left|\frac{1}{2}\,\underline{x}^T(t_f)X_f\underline{x}(t_f)\right|$$

$$+ \frac{1}{2}\int_{t_o}^{t_f}\left\{\begin{bmatrix}\underline{x}(t)\\ \underline{u}(t)\end{bmatrix}^T \begin{bmatrix}W_{xx}(t) & W_{xu}(t)\\ W_{xu}^T(t) & W_{uu}(t)\end{bmatrix} \begin{bmatrix}\underline{x}(t)\\ \underline{u}(t)\end{bmatrix}\right\} dt \tag{2-3}$$

where $W_{xx}(t)$ is a positive semi-definite weighting matrix associated with the system states, $\underline{x}(t)$; $W_{uu}(t)$ is a positive definite weighting matrix associated with the controls, $\underline{u}(t)$, applied, to the system; $W_{xu}(t)$ is a cost-weighting matrix associated with cross terms of $\underline{x}(t)$ and $\underline{u}(t)$ and is chosen so that the composite matrix of Equation (2-3) is positive semi-definite; and X_f is a positive semi-definite weighting matrix associated with the states at the final time, $\underline{x}(t_f)$.

The optimal control to be applied at time t which minimizes the above cost function is described as

$$\underline{u}^*(t) = - G_c^*(t)\ \underline{x}(t) \tag{2-4}$$

with the gain matrix, $G_c^*(t)$, given by

$$G_c^*(t) = W_{uu}^{-1}(t)\ B^T(t)\ K_c(t) \tag{2-5}$$

and $K_c(t)$ is calculated using the backward Riccati differential equation

$$\begin{aligned} -\dot{K}_c(t) = {} & F^T(t)\ K_c(t) + K_c(t)\ F(t) + W_{xx}(t) \\ & - K_c(t)\ B(t)\ W_{uu}^{-1}(t)\ B^T(t)\ K_c(t) \end{aligned} \tag{2-6}$$

subject to the final condition

$$K_c(t_f) = X_f \tag{2-7}$$

Notice that Equation (2-6) assumes that the cross weighting matrix $W_{xu}(t)$ is zero. If this is not the case, an appropriate variable transformation can be made which will account for the non-zero cross terms and allow equation (2-6) to be used (Ref 24:202-203; 19:79-86).

Up to this point, the controller equations have been derived allowing for time-varying systems, cost-weighting matrices and feedback gains. However, as described in Chapter V, the model used for this thesis is time-invariant with stationary noises (i.e., the covariance kernel

$E\{w(t)w^T(t+\tau)\}$ is a function of only the time difference τ (Ref 23:139-140)) and constant weighting matrices. Thus a steady-state constant gain controller can be used, ignoring terminal transients in the feedback gain

$$\underline{u}^*(t) = -G_c^*\underline{x}(t) = -\left[W_{uu}^{-1}B^T\bar{K}_c\right]\underline{x}(t) \qquad (2\text{-}8)$$

where $\bar{K}_c$ is now the solution to the steady-state Riccati equation

$$-\dot{\bar{K}}_c = 0 = F^T\bar{K}_c + \bar{K}_cF + W_{xx} - \bar{K}_cBW_{uu}^{-1}B^T\bar{K}_c \qquad (2\text{-}9)$$

Henceforth, the assumption of a time-invariant system with stationary noises and constant weighting matrices is made, and the time argument is omitted unless needed to avoid confusion.

The optimal control law of Equation (2-4) is given as a gain matrix times the state values at a particular time. If, however, perfect knowledge of the states is not available, then an estimate of the state, $\hat{\underline{x}}(t)$, must be used in place of $\underline{x}(t)$. In this case, a Kalman filter is employed to evaluate the conditional mean, $\hat{\underline{x}}(t)$, and conditional covariance, P(t), of the states of the system.

Instead of the actual values of the states, assume that what are accessible from the system are noise-corrupted measurements of the time-invariant form

$$\underline{z}(t) = H\,\underline{x}(t) + \underline{v}(t) \qquad (2\text{-}10)$$

where $\underline{v}(t)$ is a stationary white Gaussian noise assumed independent of the dynamics driving noise $\underline{w}(t)$ and with statistics

$$E\{\underline{v}(t)\} = \underline{0} \qquad (2\text{-}11a)$$

$$E\{\underline{v}(t)\,\underline{v}^T(t+\tau)\} = R\delta(\tau) \qquad (2\text{-}11b)$$

Then the Kalman filter equations yield an estimate of the states and the covariance via (Ref 23:257,259)

$$\dot{\hat{\underline{x}}}(t) = F\hat{\underline{x}}(t) + B\underline{u}(t) + PH^{T}R^{-1}\,[\underline{z}(t) - H\hat{\underline{x}}(t)] \tag{2-12a}$$

$$\dot{P}(t) = FP(t) + P(t)F^{T} + GQG^{T} - P(t)\,H^{T}R^{-1}\,HP(t) \tag{2-12b}$$

These differential equations are solved forward in time subject to the initial conditions

$$E\{\underline{x}(t_o)\} = \bar{\underline{x}}_o \tag{2-13a}$$

$$E\{[\underline{x}(t_o) - \bar{\underline{x}}_o]\,[\underline{x}(t_o) - \bar{\underline{x}}_o]^{T}\} = P_o \tag{2-13b}$$

which are obtained from an a priori Gaussian density function for the states at the initial time.

The precomputable Kalman filter gain for the continuous-time system is expressed in Equation (2-12a) as

$$K(t) = P(t)\,H^{T}R^{-1} \tag{2-14}$$

If it is assumed that the initial transients of Equation (2-12b) are short compared to the total time of interest, the steady-state covariance, $\bar{P}$, can be computed as the solution to

$$\dot{\bar{P}}(t) = 0 = F\bar{P} + \bar{P}F^{T} + GQG^{T} - \bar{P}H^{T}R^{-1}\,H\bar{P} \tag{2-15}$$

and the constant Kalman filter gain is given by

$$K = \bar{P}H^{T}R^{-1} \tag{2-16}$$

Do not confuse the Kalman Filter gain, K, with K_c of Equation (2-6) and (2-9).

Figure (2-1) shows the form of the continuous-time controller.

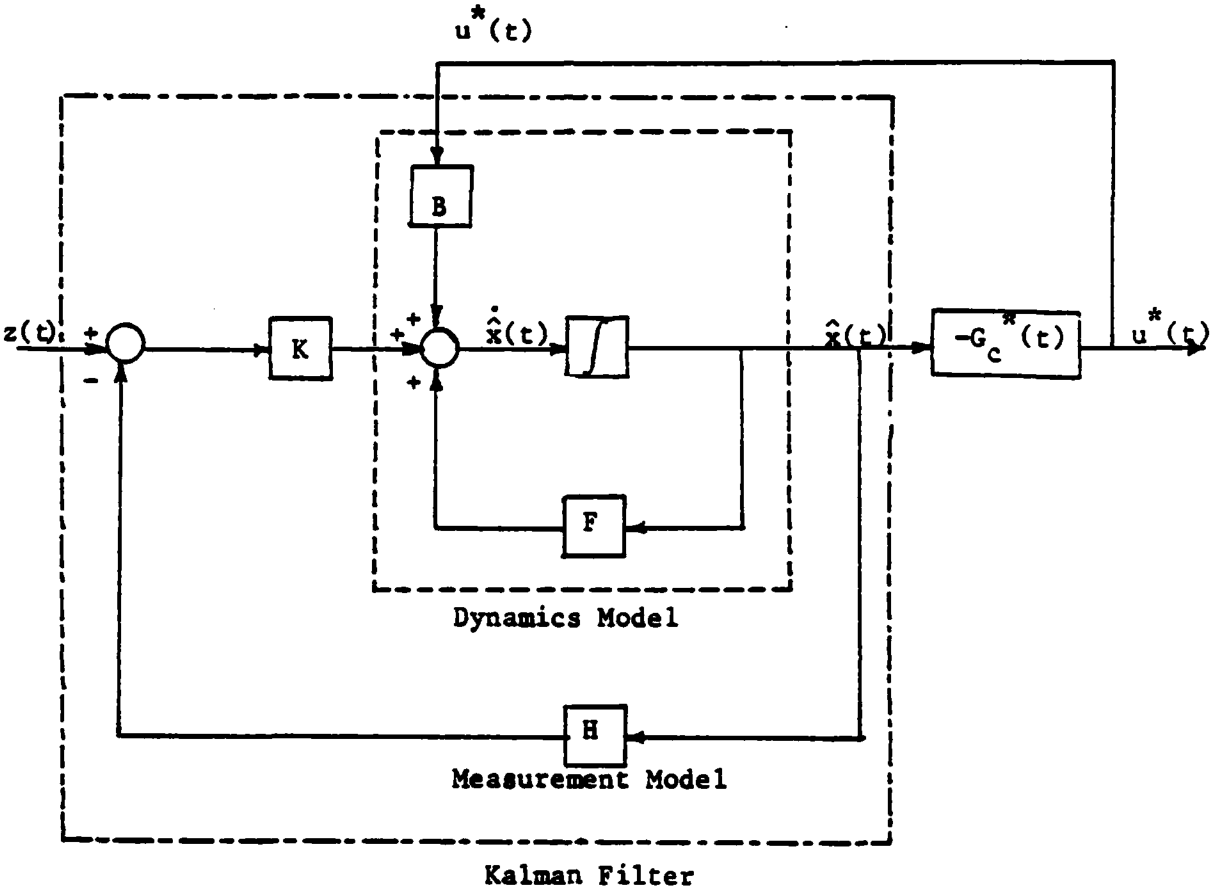


Figure 2-1: Continuous Measurement LQG Optimal Stochastic Controller

2.3 Continuous-Time Performance Analysis

The performance analysis for a continuous-time LQG controller is based on the development for a sampled-data controller given in Reference 23 and derived in Reference 21.

First, a "truth model" is developed for a given system which is judged to represent adequately the response of a system to inputs and disturbances encountered in the real world. Then, controllers for the system can be designed, generally based on lower order, simplified models. The performance is determined by examining the statistical characteristics of the truth model states, $\underline{x}_t(t)$, and the controls, $\underline{u}(t)$, for each proposed controller inserted into the real world simulation provided by the truth model. This is depicted in Figure (2-2).

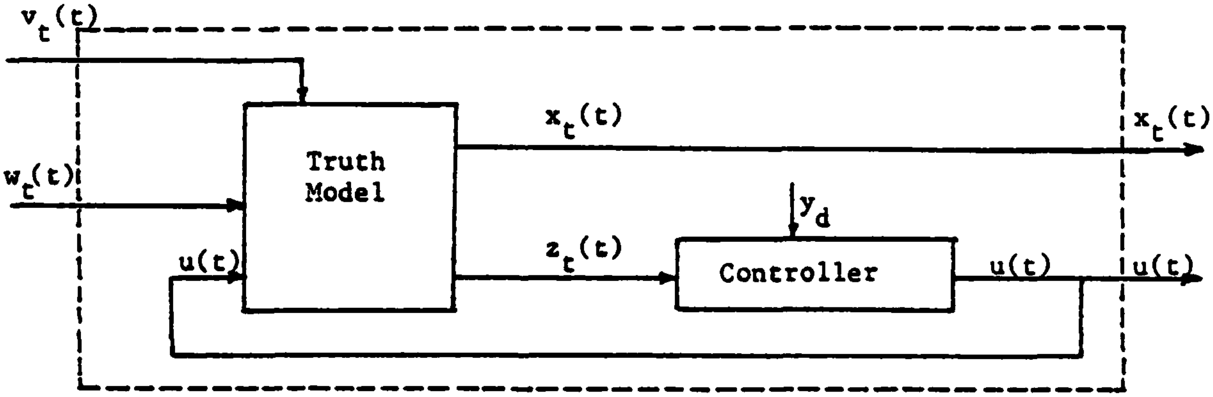


Figure 2-2: Performance Evaluation for Linear Sampled-Data Controller

The performance evaluation determines the statistical characteristics of $\underline{x}_t(t)$, the truth model states, rather than the states of the simplified controller model. It is important to determine the effect of applying controls from a reduced order controller on the actual system response.

Additionally, the statistical characteristics of the controls are examined to ensure that they do not exceed physical limits or design specifications.

The assumed linear differential equation describing the continuous-time truth model system and available measurements are

$$\dot{\underline{x}}_t(t) = F_t\underline{x}(t) + B_t\underline{u}(t) + G_t\underline{w}_t(t) \tag{2-17}$$

$$\underline{z}_t(t) = H_t\underline{x}(t) + \underline{v}_t(t) \tag{2-18}$$

where initial conditions and statistics of the noise are given by

$$E\{\underline{x}_t(t_o)\} = \bar{\underline{x}}_{to} \tag{2-19}$$

$$E\{[\underline{x}_t(t_o) - \bar{\underline{x}}_{to}][\bar{\underline{x}}_t(t_o) - \bar{\underline{x}}_{to}]^T\} = P_{to} \tag{2-20}$$

$$E\{\underline{w}_t(t)\} = 0 \tag{2-21}$$

$$E\{\underline{w}_t(t)\ \underline{w}_t^T(t+\tau)\} = Q_t\delta(\tau) \tag{2-22}$$

$$E\{\underline{v}_t(t)\} = \underline{0} \tag{2-23}$$

$$E\{\underline{v}_t(t)\ \underline{v}_t^T(t+\tau)\} = R_t\delta(\tau) \tag{2-24}$$

Note that $\underline{v}_t(t)$ and $\underline{w}_t(t)$ are assumed to be independent of each other and that the subscript t refers to the truth model.

A useful form of expressing the control law of Equation (2-4) and other more general linear control algorithms, and an equation to propagate the internal states of the controller, are given by

$$\underline{u}(t) = G_{cx}\underline{x}_c(t) + G_{cz}\,\underline{z}_t(t) + G_{cy}\,\underline{y}_d(t) \tag{2-25}$$

$$\dot{\underline{x}}_c(t) = F_c\, \underline{x}_c(t) + B_{cz}\, \underline{z}_t(t) = B_{cy}\, \underline{y}_d(t) \qquad (2\text{-}26)$$

where $\underline{y}_d(t)$ refers to a command input for the system controlled variable, $\underline{y}(t)$, to track, where

$$\underline{y}(t) = C\underline{x}(t) + D_y\, \underline{u}(t) \qquad (2\text{-}27)$$

The gain matrices in Equations (2-25) and (2-26) are evaluated explicitly in Appendix A. Putting the control law into the generic form allows direct comparison of different types of linear control laws.

Determination of time histories are desired of the mean and covariance of an augmented vector

$$\underline{y}_a(t) = \begin{bmatrix} \underline{x}_t(t) \\ \\ \underline{u}(t) \end{bmatrix} \qquad (2\text{-}28)$$

All quantities of interest in a performance analysis are assumed to be components or linear combinations of components of this vector. If q_k is a scalar quantity of interest, i.e.,

$$q_k = \underline{q}_k^T\, \underline{y}_a(t) \qquad (2\text{-}29)$$

then the statistics of q_k are given by

$$\text{mean}\{q_k(t)\} = \underline{q}_k^T\, \underline{m}_{y_a}(t) \qquad (2\text{-}30a)$$

$$\text{cov}\{\, q_k(t)\} = \underline{q}_k^T\, P_{y_a y_a}(t)\, \underline{q}_k \qquad (2\text{-}30b)$$

where $\underline{m}_{y_a}(t)$ is the mean and $P_{y_a y_a}(t)$ is the covariance of the augmented vector $\underline{y}_a(t)$.

To evaluated the first two moments of $\underline{y}_a(t)$, it is first necessary to form another augmented vector composed of the internal states of the truth model and the controller model

$$\underline{x}_a(t) = \begin{bmatrix} \underline{x}_t(t) \\ \\ \underline{x}_c(t) \end{bmatrix} \qquad (2\text{-}31)$$

Using Equation (2-18) and (2-25), $\underline{u}(t)$ and $\underline{z}_t(t)$ are eliminated from Equation (2-17)

$$\dot{\underline{x}}_t(t) = \{ F_t + B_t G_{cz} H_t \} \underline{x}_t(t) + B_t G_{cx} \underline{x}_c(t)$$

$$+ B_t G_{cy} \underline{y}_d(t) + B_t G_{cz} \underline{v}_t(t) + G_t \underline{w}_t(t) \qquad (2\text{-}32)$$

Similar substitution into Equation (2-26) yields

$$\dot{\underline{x}}_c(t) = F_c \underline{x}_c(t) + B_{cy} \underline{y}_d(t) + B_{cz} [H_t \underline{x}_t(t) + \underline{v}_t(t)] \qquad (2\text{-}33)$$

Form the augmented noise vector

$$\underline{w}_a(t) = \begin{bmatrix} \underline{w}_t(t) \\ \\ \underline{v}_t(t) \end{bmatrix} \qquad (2\text{-}34)$$

as a zero-mean white Gaussian noise with covariance kernel $E\{w_a(t) w_a^T(t + \tau)\} = Q_a \delta(\tau)$, where

$$Q_a = \begin{bmatrix} Q_t & 0 \\ 0 & R_t \end{bmatrix} \qquad (2\text{-}35)$$

Non-zero cross terms can easily handle the case of $\underline{w}_t(t)$ and $\underline{v}_t(t)$ being correlated. Now an augmented system equation can be written as

$$\dot{\underline{x}}_a(t) = F_a\underline{x}_a(t) + B_a\underline{y}_d(t) + G_a\underline{w}_a(t) \qquad (2\text{-}36)$$

where

$$F_a = \begin{bmatrix} F_t + B_t\ G_{cz}H_t & B_tG_{cx} \\ B_{cz} & F_c \end{bmatrix} \qquad (2\text{-}37)$$

$$B_a = \begin{bmatrix} B_tG_{cy} \\ B_{cy} \end{bmatrix} \qquad (2\text{-}38)$$

$$G_a = \begin{bmatrix} G_t & B_tG_{cz} \\ 0 & B_{cz} \end{bmatrix} \qquad (2\text{-}39)$$

The initial conditions for the augmented vector are

$$E\{\underline{x}_a(t_o)\} = \begin{bmatrix} \bar{\underline{x}}_{to} \\ \bar{\underline{x}}_{co} \end{bmatrix} = \bar{\underline{x}}_{ao} \qquad (2\text{-}40)$$

$$P_a(t_o) = \begin{bmatrix} P_o & 0 \\ & \\ 0 & 0 \end{bmatrix} \qquad (2\text{-}41)$$

Initial conditions on the controller states are often assumed to be zero: $\bar{\underline{x}}_{co} = \underline{0}$.

The mean and covariance of the states in Equation (2-36) propagate as

$$\dot{\underline{m}}_{x_a}(t) = F_a \underline{m}_{x_a}(t) + B_a \underline{y}_d(t) \qquad (2\text{-}42)$$

$$\dot{P}_{x_a x_a}(t) = F_a P_{x_a x_a}(t) + P_{x_a x_a}(t) F_a^T + G_a Q_a G_a^T \qquad (2\text{-}43)$$

or solving the differential equation yields

$$\underline{m}_{x_a}(t) = \phi_a(t,t_o)\bar{\underline{x}}_{ao} + \int_{t_o}^{t} \phi_a(t,\tau) B_a \underline{y}_d(\tau)\, d\tau \qquad (2\text{-}44)$$

$$P_{x_a x_a}(t) = \phi_a(t,t_0) P_{ao} \phi_a^T(t,t_0)$$

$$+ \int_{t_o}^{t} \phi_a(t,\tau) G_a Q_a G_a^T \phi_a^T(t,\tau)\, d\tau \qquad (2\text{-}45)$$

where $\phi_a(t,t_o)$ is the state transition matrix associated with F_a, i.e., $\phi_a(t,t_o) = \phi_a(t-t_o) = \exp\{F_a(t-t_o)\}$.

However, the requirement is to solve for the statistics of the vector $\underline{y}_a(t)$ defined in Equation (2-32). This vector is related to $\underline{x}_a(t)$ by

$$\underline{y}_a(t) = \begin{bmatrix} I & 0 \\ G_{cz}H_t & G_{cx} \end{bmatrix} \begin{bmatrix} \underline{x}_t(t) \\ \underline{x}_c(t) \end{bmatrix} + \begin{bmatrix} 0 \\ G_{cy} \end{bmatrix} \underline{y}_d(t) + \begin{bmatrix} 0 \\ G_{cz} \end{bmatrix} \underline{v}_t(t) \qquad (2\text{-}46)$$

The vector is seen to be a linear combination of jointly Gaussian variables with known statistics. Thus, the desired statistics can be generated from Equation (2-39) using the method shown in Reference 23:112.

$$\underline{m}_{y_a}(t) = \begin{bmatrix} I & 0 \\ G_{cz}H_t & G_{cx} \end{bmatrix} \underline{m}_{x_a}(t) + \begin{bmatrix} 0 \\ G_{cy} \end{bmatrix} \underline{y}_d(t) \qquad (2\text{-}47)$$

$$\begin{aligned} P_{y_a y_a}(t) = {} & \begin{bmatrix} I & 0 \\ G_{cz}H_t & G_{cx} \end{bmatrix} P_{x_a x_a}(t) \begin{bmatrix} I & 0 \\ G_{cz}H_t & G_{cx} \end{bmatrix}^T \\ & + \begin{bmatrix} I & 0 \\ G_{cz}H_t & G_{cx} \end{bmatrix} P_{x_a v_t}(t) \begin{bmatrix} 0 & G_{cz}^T \end{bmatrix} \\ & + \begin{bmatrix} 0 \\ G_{cz} \end{bmatrix} P_{x_a v_t}^T(t) \begin{bmatrix} I & 0 \\ G_{cz}H_t & G_{cx} \end{bmatrix}^T \\ & + \begin{bmatrix} 0 \\ G_{cz} \end{bmatrix} P_{v_t v_t}(t) \begin{bmatrix} 0 & G_{cz}^T \end{bmatrix} \qquad (2\text{-}48) \end{aligned}$$

In the last term, note that $P_{v_t v_t}(t) = R_t \delta(0)$ and that $\delta(0) = \infty$. This terms gives an infinite value to the lower right-hand partition of the covariance matrix. However, it is still of interest to look at the contribution of the other three terms separately. To evaluate Equation (2-48) fully, it is necessary to find an expression for $P_{x_a v_t}(t)$. This can be written as

$$P_{x_a v_t} = E\{[\underline{x}_a(t) - \underline{m}_{x_a}(t)][\underline{v}_t(t) - \underline{m}_{v_t}(t)]^T\} \tag{2-49}$$

Expand this, noting that $\underline{m}_{v_t}(t) = 0$

$$P_{x_a v_t}(t) = E\{\underline{x}_a(t)\underline{v}_t{}^T(t) - \underline{m}_{x_a}(t)\underline{v}_t{}^T(t)\} \tag{2-50}$$

$$p_{x_a v_t} = E\,\{\underline{x}_a(t)\underline{v}_t{}^T(t)\} - \underline{m}_{x_a}(t)E\,\{\underline{v}_t{}^T(t)\} \tag{2-51}$$

but $E\{\underline{v}_t^T(t)\} = \underline{m}_{v_t}{}^T(t) = \underline{0}^T$, therefore

$$P_{x_a v_t}(t) = E\,\{\underline{x}_a(t)\underline{v}_t{}^T(t)\} \tag{2-52}$$

Replace $\underline{x}_a(t)$ with the solution form for Equation (2-36)

$$\begin{aligned} P_{x_a v_t}(t) = E\{&\phi_a(t,t_o)\underline{x}_a(t_o)\underline{v}_t{}^T(t) \\ &+ \int_{t_o}^{t} \phi_a(t,\tau)B_a\underline{y}_d(\tau)\ \underline{v}_t{}^T(t)\ d\tau \\ &+ \int_{t_o}^{t} \phi_a(t,\tau)G_a\underline{w}_a(\tau)\ \underline{v}_t{}^T(t)\ d\tau\} \end{aligned} \tag{2-53}$$

Making the assumption that $\underline{x}_a(t_o)$ and $\underline{v}_t(t)$ are independent (and thus uncorrelated), the first term in Equation (2-53) is zero since $E\{v_t^T(t)\} = \underline{0}^T$. Substituting in the augmented matrices, the remaining terms are

$$P_{x_a v_t}(t) = E\left\{\int_{t_o}^{t} \phi_a(t,\tau) \begin{bmatrix} B_t & G_{cy} \\ & \\ & B_{cy} \end{bmatrix} \underline{y}_d(\tau)\ \underline{v}_t^T(t)d\tau\right.$$

$$\left. + \int_{t_o}^{t} \phi_a(t,\tau) \begin{bmatrix} G_t & B_tG_{cz} \\ & \\ 0 & B_{cz} \end{bmatrix} \begin{bmatrix} \underline{w}_t(\tau) \\ \\ \underline{v}_t(\tau) \end{bmatrix} \underline{v}_t^T(t)d\tau \right\} \qquad (2\text{-}54)$$

which can be rewritten as

$$P_{x_a v_t}(t) = E\left\{\int_{t_o}^{t} \phi_a(t,\tau) \begin{bmatrix} B_tG_{cy} \\ \\ B_{cy} \end{bmatrix} \underline{y}_d(\tau)\ \underline{v}_t^T(t)d\tau\right.$$

$$\left. + \int_{t_o}^{t} \phi_a(t,\tau) \begin{bmatrix} G_t\underline{w}_t(\tau) + B_tG_{cz}\underline{v}_t(\tau) \\ \\ B_{cz}\underline{v}_t(\tau) \end{bmatrix} \underline{v}_t^T(t)d\tau \right\} \quad (2\text{-}55)$$

Again, the first term is zero since $\underline{v}_t(t)$ is zero-mean, and $\underline{y}_d(\tau)$ is deterministic. This leaves

$$P_{x_a v_t}(t) = E\left\{\int_{t_o}^{t} \phi_a(t,\tau) \begin{bmatrix} G_t \underline{w}_t(\tau) + B_t G_{cz} \underline{v}_t(\tau) \\ \\ B_{cz} \underline{v}_t(\tau) \end{bmatrix} \underline{v}_t^T(t) d\tau\right\} \qquad (2\text{-}56)$$

Noting that $\underline{w}_t(\tau)$ and $\underline{v}_t(t)$ are assumed independent and

$$E\{\underline{v}_t(\tau)\underline{v}_t^T(t)\} = R_t \delta(t-\tau) \qquad (2\text{-}57)$$

the expected value operation can be moved inside the integral, and Equation (2-56) further reduces to

$$P_{x_a v_t}(t) = \int_{t_o}^{t} \phi_a(t_, \tau) \begin{bmatrix} B_t G_{cz} \\ \\ B_{cz} \end{bmatrix} R_t \delta(t-\tau) d\tau \qquad (2\text{-}58)$$

Applying the Dirac delta sifting property to the above equation, noting that t is the upper limit of the integration and that it also appears in the delta function argument, yields

$$P_{x_a v_t}(t) = \phi_a(t,t) \begin{bmatrix} B_t G_{cz} \\ \\ B_{cz} \end{bmatrix} \frac{1}{2} R_t \qquad (2\text{-}59)$$

The state transition matrix $\phi_a(t,t)$ is the identity matrix I. The value of $\frac{1}{2}$ appears when integrating the Dirac delta function between the time limits t_o and t since its argument goes to zero at $\tau = t$, i.e., at its upper limit. The resulting form is

$$P_{x_a v_t} = \frac{1}{2} \begin{bmatrix} B_t G_{cz} \\ \\ B_{cz} \end{bmatrix} R_t \qquad (2\text{-}60)$$

At this point, all quantities needed to obtain the performance analysis for a linear, time-invariant, continuous-time system and controller have been defined in this section. The primary result is that, for a truth model given by Equations (2-17) and (2-24) and a controller of the form (2-25) and (2-26), the statistics of desired outputs are given by (2-47) and (2-48), using (2-42) through (2-45) and (2-60).

2.4 Improving Robustness in Continuous-Time Controllers

The stability robustness of LQG controllers is guaranteed assuming that full-state feedback is available. However, once a Kalman filter is inserted into the loop, all guarantees of robustness, such as minimum gain and phase margins, are lost (Ref 6). J.C. Doyle demonstrated this for a simple case of an observer-based controller in Reference 5.

In Reference 6, Doyle and Stein introduced a method for improving the robustness of a control system that employs an observer or state estimator to generate estimates of states when full-state feedback is unavailable. In many practical systems, this is the case. Their method assumes that the linear, time-invariant system to be controlled is observable, controllable and has no transmission zeros in the right-half s-plane, (i.e., it is minimum phase).

Figure (2-3) shows the structures for a full-state feedback controller and an observer-based controller. Doyle and Stein claim that if

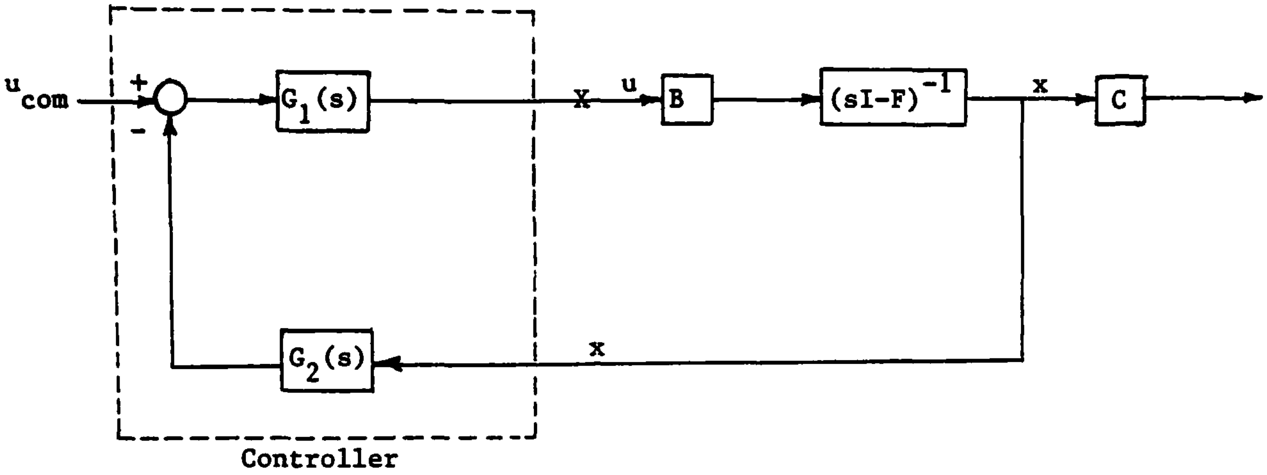


Figure 2-3a: Full-State Feedback System

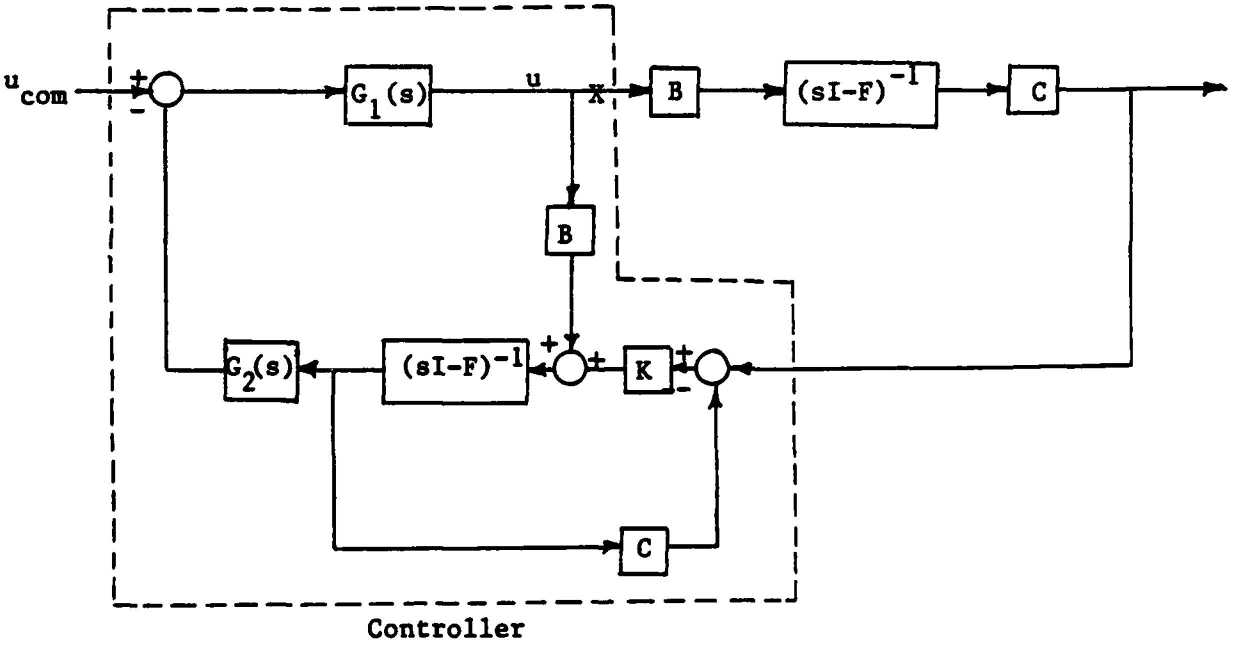


Figure 2-3b: Observer Based Feedback System

the corresponding return-difference mappings are asymptotically equal when the control loops are broken at point x (the point of entry of the control inputs), then the robustness properties cf the observer-based controller will asymptotically approach those of the full-state feedback controller.

The return-difference mappings of Figure (2-3) are equal if the observer satisfies the following equation

$$K\left[I + H(sI-F)^{-1}K\right]^{-1} = B\left[H(sI-F)^{-1}B\right]^{-1} \qquad (2\text{-}61)$$

where H,F, and B are system matrices and K is the observer gain. Let K be parameterized as a function of the scalar variable, q. Equation (2-61) is satisfied asymptotically as q approaches infinity if

$$\frac{K(q)}{q} \text{ ----> } BW \qquad (2\text{-}62)$$

where W is any nonsingular matrix. If the observer used is Kalman filter, then K(q) becomes the Kalman filter gain

$$K(q) = \bar{P}(q)H^{T}R^{-1} \qquad (2\text{-}63)$$

and $\bar{P}(q)$ replaces in Equation (2-16).

To implement the Doyle and Stein technique, the value of GQG^T in Equation (2-16) must be altered. Let Q_o be the matrix GQG^T of the original system (i.e., let G = I) and O(q) be the modified matrix after the robustification technique is applied. The modified matrix is

$$Q(q) = Q_o + q^2BVB^T \qquad (2\text{-}64)$$

where q is the design parameter chosen to reflect the amount of desired robustification and V is a positive definite symmetric matrix. Note that the second term of Equation (2-64) implies adding additional pseudonoise to the system at the point of entry of the control inputs, $\underline{u}(t)$, rather than the point of entry of the dynamic driving noise, $\underline{w}(t)$. As q approaches infinity, the observer-based controller recovers the robustness properties of a full-state feedback controller. Note that if q is zero, Q(q) is the GQG^T matrix of the original system.

2.5 Sampled-Data Controller

For a continuous-time system, represented by Equation (2-1), having sampled-data measurements, an equivalent discrete-time stochastic difference equation (Ref 23:170) can be written as

$$\underline{x}(t_{i+1}) = \phi(t_{i+1},t_i)\ \underline{x}(t_i) + B_d(t_i)\ \underline{u}(t_i) + G_d(t_i)\ \underline{w}_d(t_i) \qquad (2\text{-}65)$$

where $\underline{w}_d(t_i)$ is a discrete-time white Gaussian noise with statistics

$$E\{\underline{w}_d(t_i)\} = \underline{0} \qquad (2\text{-}66a)$$

$$E\{\underline{w}_d(t_i)\ \underline{w}_d^T(t_j)\} = Q_d(t_i)\delta_{ij} \qquad (2\text{-}66b)$$

and δ_{ij} is the Kronecker delta, equal to one if i=j and zero if i≠ j.

The matrix $\phi(t_{i+1},t_i)$ is the state transition matrix for the system over a single sample period and $B_d(t_i)$ and $Q_d(t_i)$ are defined in terms of the continuous-time system matrices to be

$$B_d(t_i) = \int_{t_i}^{t_{i+1}} \phi(t_{i+1},\tau)\ B(\tau)\ d\tau \qquad (2\text{-}67)$$

$$Q_d(t_i) = \int_{t_i}^{t_{i+1}} \phi(t_{i+1},\tau)\ G(\tau)\ Q(\tau)\ G^T(\tau)\phi^T(t_{i+1},\tau)d\tau \qquad (2\text{-}68)$$

Note that Equation (2-67) inherently assumes $\underline{u}(t)$ is held constant over a sample period, i.e., $\underline{u}(t) = \underline{u}(t_i)$ for all t_i in the interval $[t_i,t_{i+1})$. $G_d(t_i)$ is defined to be an identity matrix I. If, as in the continuous-time controller case, a time invariant system model and stationary noises are assumed, and if in addition a fixed sample period is assumed, the integrations of Equation (2-67) and (2-68) need only be performed once and B_d and Q_d are constant matrices.

A discrete-time cost function similar to that of Equation (2-3) can be generated as

$$J_c = E\left\{\frac{1}{2}\underline{x}^T(t_{n+1})\ X_f\ \underline{x}(t_{n+1}) + \sum_{i=0}^{n}\frac{1}{2}\begin{bmatrix}\underline{x}(t_i)\\ \\ \underline{u}(t_i)\end{bmatrix}^T\begin{bmatrix}X(t_i) & S(t_i)\\ \\ S^T(t_i) & U(t_i)\end{bmatrix}\begin{bmatrix}\underline{x}(t_i)\\ \\ \underline{u}(t_i)\end{bmatrix}\right\} \qquad (2\text{-}69)$$

In the above equation, t_n is the last time at which a control is applied. It is assumed that a zero-order-hold is used to interface with the continuous-time system. X_f and $X(t_i)$ are positive semi-definite weighting matrices and $U(t_i)$ is a positive definite weighting matrix. $S(t_i)$ is chosen so that the composite matrix of Equation (2-69) is positive semi-definite. The weighting matrices are again assumed constant henceforth, in order to obtain constant gain control laws eventually.

The LQ optimal full-state feedback control law is given by

$$u^*(t_i) = -G_c^*(t_i)\,\underline{x}(t_i) \qquad (2\text{-}70)$$

where $G_c^*(t_i)$ is the solution to the equation

$$G_c^*(t_i) = [U + B_d^T\, K_c(t_{i+1})\, B_d]^{-1}\, [B_d^T\, K_c(t_{i+1})\, \phi + S^T] \qquad (2\text{-}71)$$

and $K_c(t_i)$ satisfies the backward recursive Riccati difference quation

$$K_c(t_i) = X + \phi^T K_c(t_{i+1})\,\phi - [B_d^T K_c(t_{i+1})\,\phi + S^T]^T\, G_c^*(t_i) \qquad (2\text{-}72)$$

subject to the final condition

$$K_c(t_{n+1}) = X_f \qquad (2\text{-}73)$$

As in the case of a continuous-time system having continuous-time measurements, perfect access to all the states usually is not available. Instead, sampled-data measurements may be available in the form of

$$\underline{z}(t_i) = H\,\underline{x}(t_i) + \underline{v}_d(t_i) \qquad (2\text{-}74)$$

where $\underline{v}_d(t_i)$ is a discrete-time white Gaussian noise assumed independent of dynamics driving noise $\underline{w}_d(t_i)$ in Equation (2-65), with statistics

$$E\,\{\underline{v}_d(t_i)\} = \underline{0} \qquad (2\text{-}75a)$$

$$E\{\underline{v}_d(t_i)\,\underline{v}_d^T(t_j)\} = R_d\ \ \delta_{ij} \qquad (2\text{-}75b)$$

Again, a Kalman filter is employed to generate an estimate of the conditional mean, $\hat{\underline{x}}(t_i+)$, and conditional covariance, $P(t_i^+)$, of the states of the system.

The mean and covariance are propagated between sampled times by these equations (Ref 23:217)

$$\hat{x}(t_{i+1}^-) = \phi\underline{x}(t_i^+) + B_d\underline{u}(t_i) \tag{2-76a}$$

$$P(t_{i+1}^-) = \phi P(t_i^+)\,\phi^T + G_d\,Q_d\,G_d^T \tag{2-76b}$$

Then, at the sample times when measurements are taken, the mean and covariance are updated by (Ref 23:217)

$$\hat{x}(t_i^+) = \hat{x}(t_i^-) + K(t_i)\,[\underline{z}(t_i) - H\,\hat{\underline{x}}(t_i^-)] \tag{2-77}$$

$$P(t_i^+) = P(t_i^-) - K(t_i)H\,P(t_i^-) \tag{2-78}$$

where $K(t_i)$ is the Kalman filter gain, given by

$$K(t_i) = P(t_i^-)\,H^T[H\,P(t_i^-)\,H^T + R_d]^{-1} \tag{2-79}$$

The superscripts, - and +, refer to quantities just before and just after the measurements at the samples times are taken, respectively.

The mean and covariance are propagated and updated forward in time by the above discrete-time Kalman filter equations subject to the initial conditions

$$E\{\underline{x}(t_o)\} = \bar{\underline{x}}_o \tag{2-80a}$$

$$E\{[\underline{x}(t_o) - \bar{\underline{x}}_o][\underline{x}(t_o) - \bar{\underline{x}}_o]^T\} = P_o \qquad (2\text{-}80b)$$

which are obtained from an a priori Gaussian density function of the states at the initial time, as discussed below Equation (2-13).

As in the continuous-measurement case, it is desired to generate a constant-gain, steady-state LQG controller. Thus, steady-state solutions from Equations (2-72), (2-76b) and (2-78) are desired to generate a constant Kalman filter gain matrix, K, and a constant feedback gain matrix, G_c^*.

The form of the sampled-data controller is shown in Figure (2-4).

2.6 Sampled-Data Performance Analysis

The performance analysis for a sampled-data LQG controller is based on the development in Reference 24:Ch 14.

Similar to Section 2.3, a truth model for a given system is developed which has the following form

$$\dot{\underline{x}}_t(t) = F_t\, \underline{x}_t(t) + B_t\, \underline{u}(t) + G_t\, \underline{w}_t(t) \qquad (2\text{-}81a)$$

$$\underline{z}_t(t_i) = H_t\, \underline{x}_t(t_i) + \underline{v}_{dt}(t_i) \qquad (2\text{-}81b)$$

The measurements are now in sampled-data form, and $\underline{v}_{dt}(t_i)$ is a discrete time white Gaussian noise of covariance R_{dt}.

As shown in Figure (2-5), proposed sampled-data controllers can be inserted into the truth model simulation of the real world and their performance evaluated.

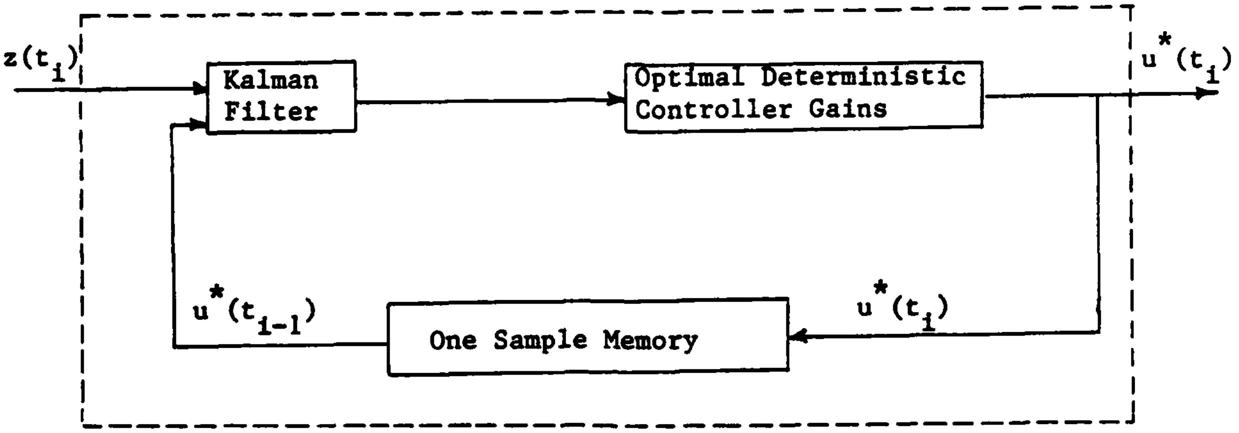


Figure 2-4a: Overall Structure of Sampled-Data LQG Regulator

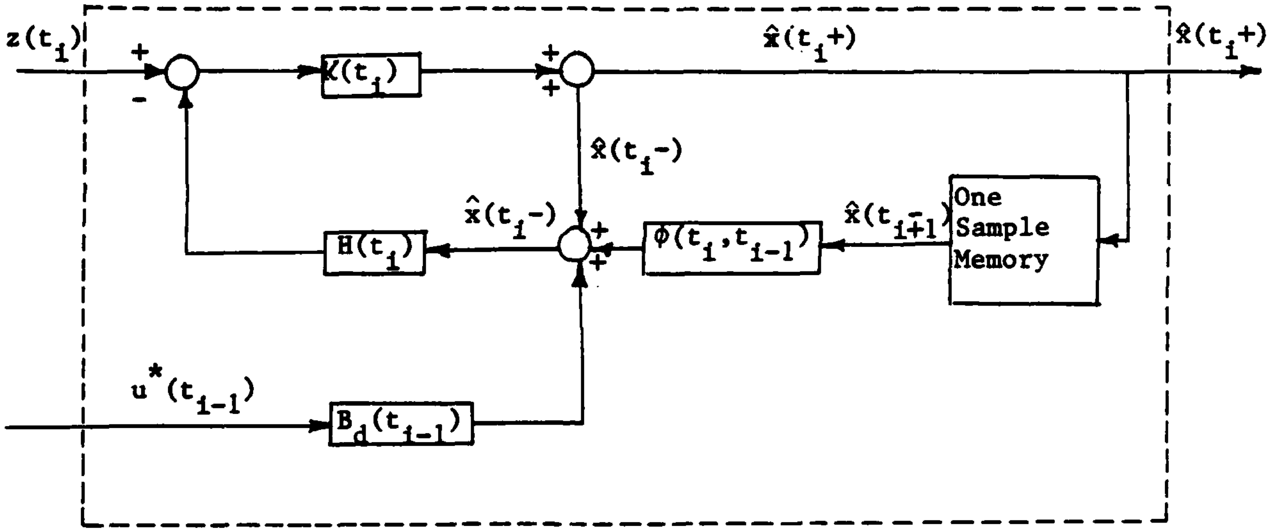


Figure 2-4b: Kalman Filter Block in (A)

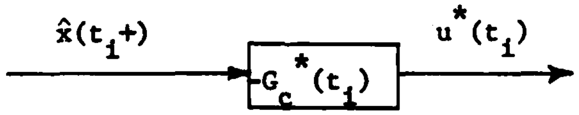


Figure 2-4c: Optimal Deterministic Controller Block in (A)

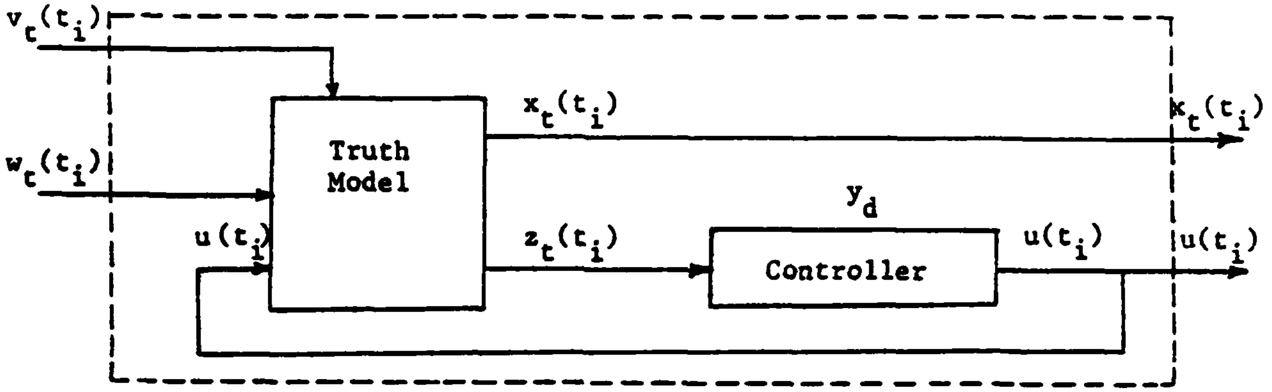


Figure 2-5: Performance Evaluation for Linear Sampled-Data Controller

For each controller, it is desired to evaluate a control law of the generic form

$$\underline{u}(t_i) = G_{cx}\,\underline{x}_c(t_i) + G_{cz}\,\underline{z}_t(t_i) + G_{cy}\,\underline{y}_d(t_i) \qquad (2\text{-}82)$$

where $\underline{u}(t_i)$ uses measurements up to and including the ith measurement and is held constant over the ith sample period. The states of the controller can be propagated by the linear difference equation

$$\underline{x}_c(t_{i+1}) = \phi_c \underline{x}_c(t_i) + B_{cz}\,\underline{z}_t(t_i) + B_{cy}\,\underline{y}_d(t_i) \qquad (2\text{-}83)$$

Note the analogous form of the above two equation to that of Equation (2-25) and (2-26). Expressions for the gains multiplying $\underline{x}_c(t_i)$, $\underline{z}_t(t_i)$ and $\underline{y}_d(t_i)$ aregiven in Appendix A. Initial conditions for the controller states are usually assumed zero.

The performance of a controller is evaluated by generating time histories of the mean and covariance of the augmented Gaussian vector

$$\underline{y}_a(t_i) = \begin{bmatrix} \underline{x}_t(t_i) \\ \\ \underline{u}(t_i) \end{bmatrix} \tag{2-84}$$

and the corresponding vector between sample times. This is developed subsequent to a description pertaining to sample times only.

First, as in Section 2.4, it is necessary to evaluate the statistics of a second augmented vector

$$\underline{x}_a(t_i) = \begin{bmatrix} \underline{x}_t(t_i) \\ \\ \underline{x}_c(t_i) \end{bmatrix} \tag{2-85}$$

By performing the integrations indicated in Equations (2-67) and (2-68), an equivalent discrete-time equation can be generated for Equation (2-81a). Then by substituting Equations (2-81b) and (2-82) into the equations for the truth model and controller states, Equations (2-81a) and (2-83) can be rewritten as

$$\begin{aligned} x_t(t_{i+1}) = {} & [\phi_t + B_{dt}G_{cz}H_t]\, \underline{x}_t(t_i) \\ & + B_{dt}G_{cx}\underline{x}_c(t_i) + B_{dt}G_{cy}\underline{y}_d(t_i) \\ & + I\, \underline{w}_{dt}(t_i) + B_{dt}G_{cz}\, \underline{v}_{dt}(t_i) \end{aligned} \tag{2-86}$$

$$\begin{aligned} \underline{x}_c(t_{i+1}) + {} & \phi_c\underline{x}_c(t_i) + B_{cz}H_t\, \underline{x}_t(t_i) \\ & + B_{cy}\, \underline{y}_d(t_i) + B_{cz}\, \underline{v}_{dt}(t_i) \end{aligned} \tag{2-87}$$

The subscript d referes to a discrete quantity,

Form the augmented, stationary, zero-mean white Gaussian noise vector

$$\underline{w}_{da}(t_i) = \begin{bmatrix} \underline{w}_{dt}(t_i) \\ \\ \underline{v}_{dt}(t_i) \end{bmatrix} \tag{2-88}$$

with covariance

$$Q_{da} = \begin{bmatrix} Q_{dt} & 0 \\ \\ 0 & R_{dt} \end{bmatrix} \tag{2-89}$$

If $\underline{w}_{dt}(t_i)$ and $\underline{v}_{dt}(t_i)$ are correlated when off-diagonal terms can be added to Equation (2-89).

Now an augmented system equation can be written by combining Equations (2-86) and (2-87)

$$\underline{x}_a(t_{i+1}) = \phi_a \underline{x}(t_i) + B_{da}\, \underline{y}_d(t_i) + G_{da}\, \underline{w}_{da}(t_i) \tag{2-90}$$

where

$$\phi_a = \begin{bmatrix} \phi_t + B_{dt}G_{cz}H_t & B_{dt}G_{cx} \\ \\ B_{cz}H_t & \phi_c \end{bmatrix} \tag{2-91}$$

$$B_{da} = \begin{bmatrix} B_{dt}G_{cy} \\ \\ B_{cy} \end{bmatrix} \tag{2-92}$$

$$G_{da} = \begin{bmatrix} I & B_{dt}G_{cz} \\ 0 & B_{cz} \end{bmatrix} \qquad (2\text{-}93)$$

The initial conditions for the augmented system are given by

$$E\{\underline{x}_a(t_o)\} = \begin{bmatrix} \bar{\underline{x}}_o \\ \underline{0} \end{bmatrix} = \bar{\underline{x}}_{ao} \qquad (2\text{-}94)$$

$$E\{[\underline{x}_a(t_o) - \bar{\underline{x}}_{ao}][\underline{x}_a(t_o) - \bar{\underline{x}}_{ao}{}^T]\} = \begin{bmatrix} P & 0 \\ 0 & 0 \end{bmatrix} \qquad (2\text{-}95)$$

If desired, a cost function for the augmented system can be formed, as shown in References 24:Ch 14;12.

The mean and covariance of the augmented vector, $\underline{x}_a(t_i)$ can be propagated by

$$\underline{m}_{x_a}(t_i) = \phi_a \underline{m}_{x_a}(t_i) + B_{da}\, \underline{y}_d(t_i) \qquad (2\text{-}96)$$

$$P_{x_a x_a}(t_i) = \phi_a P_{x_a x_a} \phi_a{}^T + G_{da} Q_{da} G_{da}{}^T \qquad (2\text{-}97)$$

The vector of interest, $\underline{y}_a(t_i)$, is related to $\underline{x}_a(t_i)$ by the following equation

$$\underline{y}_a(t_i) = \begin{bmatrix} \underline{x}_t(t_i) \\ \\ \underline{u}(t_i) \end{bmatrix} = \begin{bmatrix} I & 0 \\ \\ G_{cz}H_t & G_{cz} \end{bmatrix} \begin{bmatrix} \underline{x}_t(t_i) \\ \\ \underline{x}_c(t_i) \end{bmatrix}$$

$$+ \begin{bmatrix} 0 \\ \\ G_{cy} \end{bmatrix} \underline{y}_d(t_i) + \begin{bmatrix} 0 \\ \\ G_{cz} \end{bmatrix} \underline{v}_{dt}(t_i) \qquad (2\text{-}98)$$

and the desired statistics are therefore generated by

$$\underline{m}_{ya}(t_i) = \begin{bmatrix} I & 0 \\ \\ G_{cz}H_t & G_{cx} \end{bmatrix} \underline{m}_{x_a}(t_i) + \begin{bmatrix} 0 \\ \\ G_{cy} \end{bmatrix} \underline{y}_d(t_i) \qquad (2\text{-}99)$$

$$P_{y_a y_a}(t_i) = \begin{bmatrix} I & 0 \\ \\ G_{cz}H_t & G_{cx} \end{bmatrix} P_{x_a x_a}(t_i) \begin{bmatrix} I & 0 \\ \\ G_{cz}H_t & G_{cx} \end{bmatrix}^T$$

$$+ \begin{bmatrix} 0 \\ \\ G_{cz} \end{bmatrix} R_{dt} \, [0 \quad G_{cz}^T] \qquad (2\text{-}100)$$

Equations (2-99) and (2-100) provide an efficient means for evaluating statistics at the sample times via the equivalent discrete-time system model. However, it may be desired to obtain more complete results between sample times, to ensure that adequate sample period choice and system control have been achieved. Between sample instants, a differential equation for $\underline{y}_a(t)$ can be written as

$$\dot{\underline{y}}_a(t) = \begin{bmatrix} F_t & B_t \\ & \\ 0 & 0 \end{bmatrix} \underline{y}_a(t) + \begin{bmatrix} G_t \\ \\ 0 \end{bmatrix} \underline{w}_t(t) \tag{2-101}$$

and the mean and covariance can be propagated from t_i to t_{i+1}, where $\underline{u}(t_i)$ undergoes a step change, using the initial conditions given by Equations (2-90) and (2-91). The propagation equations are

$$\dot{\underline{m}}_{y_a}(t) = \begin{bmatrix} F_t & B_t \\ & \\ 0 & 0 \end{bmatrix} \underline{m}_{y_a}(t) \tag{2-102}$$

$$\dot{P}_{y_a y_a}(t) = \begin{bmatrix} F_t & B_t \\ & \\ 0 & 0 \end{bmatrix} P_{y_a y_a}(t) + P_{y_a y_a}(t) \begin{bmatrix} F_t & B_t \\ & \\ 0 & 0 \end{bmatrix}^T$$

$$+ \begin{bmatrix} GQG^T & 0 \\ & \\ 0 & 0 \end{bmatrix} \tag{2-103}$$

At this point, all quantities of interest have been defined to evaluate the performance of a sampled-data controller. The primary results are that for a truth model given by (2-81) and a controller of the form (2-82) and (2-83), the statistics of desired outputs at the sample times are given by (2-99) and (2-100). Between sample times the statistics are given by (2-102) and (2-103).

2.7 Improving Robustness in Discrete-Time Systems

Three approaches are taken to apply the robustification technique described in Section 2.3 to sampled-data controllers. The first is simply to discretize the LQG controller designed for the continuous-time, continuous-measurement system. The second approach is to extend the technique of adding pseudonoise to the points of entry of $\underline{u}$ to a sampled-data system by making a first order (or better) approximation to the modified Q matrix. The third is to apply the fundamental conditions under which full-state feedback characteristics are obtained asymptotically from a controller with a filter or observer in the loop.

It has been observed in Reference 21 that, unlike the continuous-measurement case, the scalar parameter q of Equation (2-64) cannot be adjusted arbitrarily upwards for the discrete-time case. Rather, there is a finite range of values for q that will robustify the Kalman filter, and beyond that range the closed-loop system is unstable.

2.7.1 Discretizing the Continuous-Time LQG Controller

The required format for the discretized controller is given by Equation (2-82) which requires values for $G_{cx}(t_i)$, $G_{cz}(t_i)$ and $G_{cy}(t_i)$, and by Equation (2-83) which requires values for $\phi_c(t_{i+1},t_i)$, $B_{cz}(t_i)$ and $B_{cy}(t_i)$. Recall that the continuous-time controller equations are expressed as

$$\underline{u}(t) = G_{cx}(t)\,\underline{x}_c(t) + G_{cx}(t)\,\underline{z}(t) + G_{cy}(t)\,\underline{y}_d(t) \qquad (2\text{-}104)$$

$$\dot{\underline{x}}_c(t) = F_c(t)\,\underline{x}_c(t) + B_{cz}(t)\,\underline{z}(t) + B_{cy}(t)\,\underline{y}_d(t) \qquad (2\text{-}105)$$

In Appendix A, it is shown that for a steady-state constant-gain LQG regulator

$$G_{cx}(t) = G_c^* \tag{2-106a}$$

$$G_{cz}(t) = 0 \tag{2-106b}$$

$$G_{cy}(t) = 0 \tag{2-106c}$$

$$F_c(t) = F - BG_c^* - KH \tag{2-106d}$$

$$G_{cz}(t) = K \tag{2-106e}$$

$$B_{cy}(t) = 0 \tag{2-106f}$$

For this problem, $G_c^*(t)$ is a constant value G_c^*, and thus the discretized control law is given by

$$\underline{u}(t_i) = -G_c^* \, \underline{x}_c(t_i) \tag{2-107}$$

To obtain a discrete propagation equation for the controller states, first-order approximations for $\phi_c(t_{i+1},t_i)$, $B_{cz}(t_i)$ and $B_{cy}(t_i)$ are obtained by

$$\phi_c(t_{i+1},t_i) = [I + F_c(t_i)\Delta t] \tag{2-108a}$$

$$B_{czd}(t_i) = B_{cz}(t_i)\Delta t \tag{2-108b}$$

$$B_{cyd}(t_i) = B_{cy}(t_i)\Delta t \qquad (2\text{-}108c)$$

Again, for this thesis, the system is assumed time invariant and F_c, B_{cz}, and B_{cy} are constant matrices. They are evaluated explicitly in Appendix A. The quantity Δt is the sample period for the sampled-data system. Higher order approximations are also possible, but if Δt is not sufficiently small compared to the transient times of the system, this basic idea breaks down anyway, so its use is confined to problems with short sample periods

In addition, a discrete approximation for $R_{dt}(t_i)$ is needed for use in the performance analysis described in Section 2.6. If the sample time is sufficiently small to allow the approximations in Equation (2-108), then it is reasonable to make a first-order approximation for the discrete measurement noise covariance $R_{dt}(t_i)$. This is given by

$$R_{dt}(t_i) = R_t(t_i)/\Delta t \qquad (2\text{-}109)$$

Notice that as Δt --> 0, the discrete-time white noise $\underline{v}_{dt}(t_i)$ of Equation (2-81b) converges to the continuous-time white noise described by (2-18) and (24) if (2-109) is satisfied throughout the limiting process.

2.7.2 Doyle and Stein Technique for Sampled-Data Systems

The second approach to enhancing robustness is to extend the Doyle and Stein technique and apply it directly to a sampled-data controller. Several ways of accomplishing this are considered.

If the sampled period of the system is sufficiently small, a first-order approximation for the discrete-time noise strength matrix $Q_d(q)$, can be made which is the discrete counterpart for Equation (2-64). This yields

$$Q_d(q) + Q_{do} + q^2 \; BVB^T \; \Delta t \tag{2-110}$$

where Q_{do} is

$$Q_{do} = Q_d(t_i) \tag{2-111}$$

which is defined in Equation (2-68). Again, this implies that $G_d = I$.

For larger sample times, a first-order approximation may not be sufficient. If this is the case, subintervalling may be considered a higher order approximation for $Q_d(q)$, such as

$$Q_d(q) = \frac{1}{2} \; [\phi Q(q) + Q(q)\phi^T] \; \Delta t \tag{2-112}$$

as given in Reference 24:172.

Another alternative is to make the return-difference mapping for the observer-based controller asympotically equal to that of the full-state feedback controller in the discrete-time case. The two configurations are shown in Figure (2-6). The observer-based system is based upon the sub-optimal control law

$$\underline{u}(t_i) = -G_c^{*} \; \hat{\underline{x}}(t_i^{-}) \tag{2-113}$$

rather than the law given by Section 2.5 using $\hat{\underline{x}}(t_i^{+})$.

To recover the robustness properties of the full-state feedback system, $[\phi K]$ must be found such that

$$\phi K[I + H(zI - \phi)^{-1}\phi K]^{-1} = B_d[H \; (zI - \phi)^{-1} \; B_d]^{-1} \tag{2-114}$$

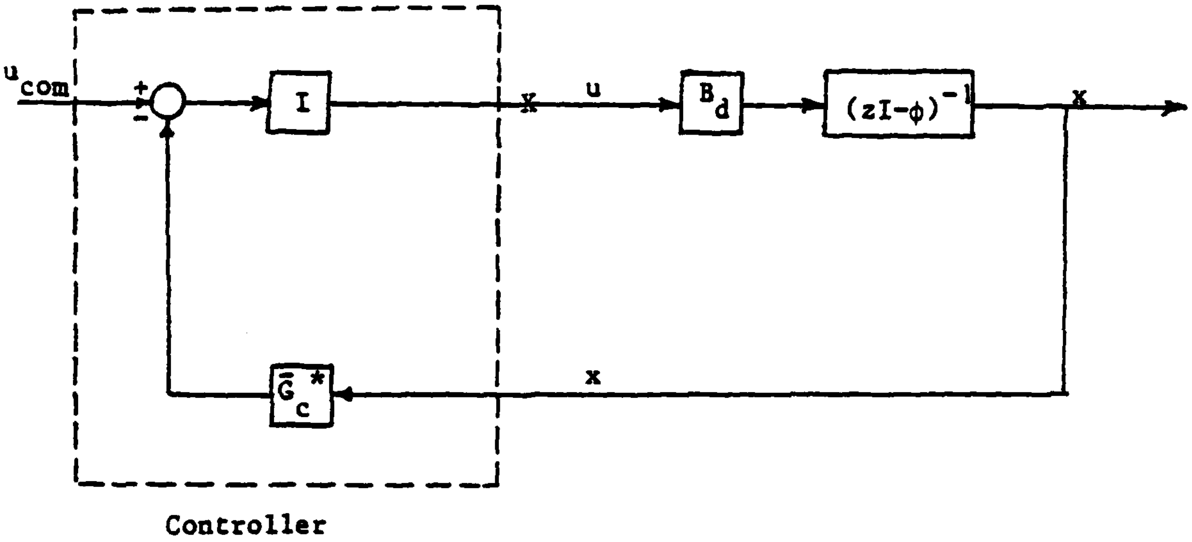


Figure 2-6a: Full-State Feedback System

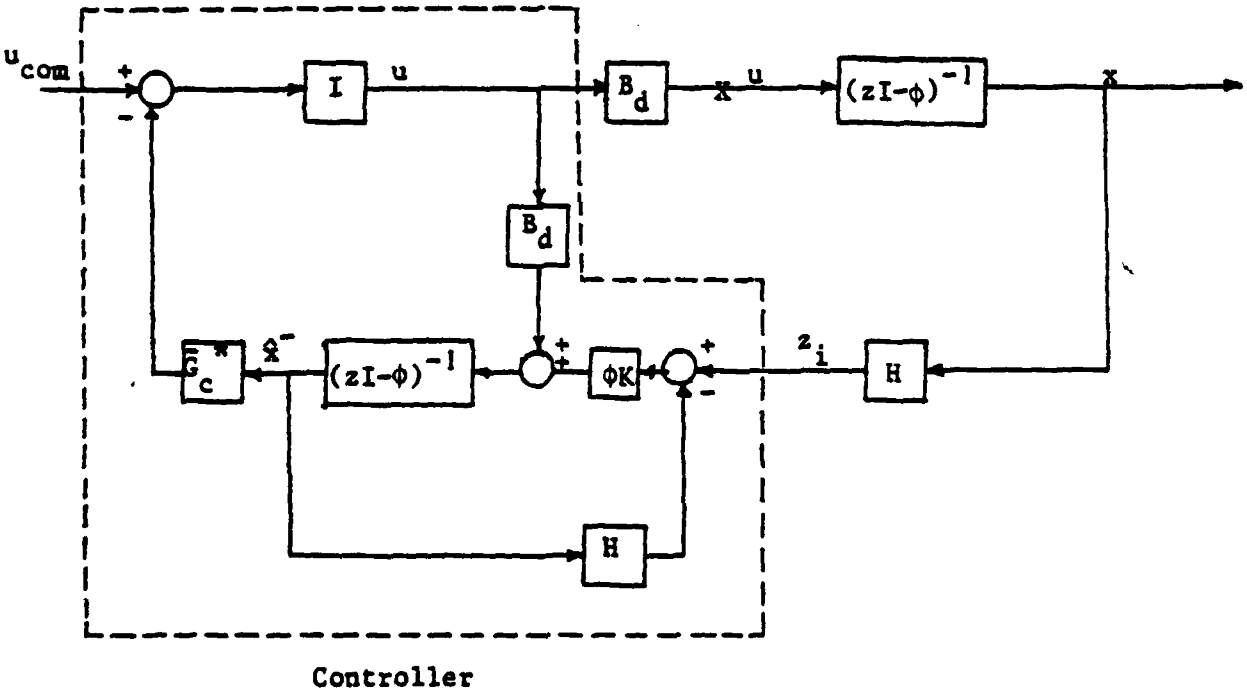


Figure 2-6b: Observer-Based Feedback System

where the analysis is carried out in the z-domain. Equation (2-114) is satisfied asymptotically when the Kalman filter gain, K, is parameterized as K(q), and

$$\lim_{q \to \infty} \frac{\phi K}{q} = B_d W \qquad (2\text{-}115)$$

where W is a nonsingular m x m matrix. Therefore, for finite values of q, K is given by

$$K = q\phi^{-1} B_d W \qquad (2\text{-}116)$$

Unfortunately, this does not lend itself to an interpretation of retuning via pseudonoise addition, as in the continuous-time case.

Reference 24:113-114 suggests looking at the dual state equations to select W. This yields

$$W = [H \phi^{-1} B_d]^{-1} \qquad (2\text{-}117)$$

which assigns m eigenvalues of the closed loop dual system to the origin and the remaining (n-m) eigenvalues to the invariant zeros of the system where n is the number of states and m is the number of controls.

To use the performance analysis equations, the sub-optimal control must be put into the generic format of Equations (2-82) and (2-83). This is shown explicitly in Appendix A.

2.8 Summary

The preceding sections have presented the equations for designing and evaluating optimal LQG regulators. Three applications are considered:

a continuous-time controller, a discretized continuous-time controller and a sampled-data controller. The structure of these controllers was examined with and without a Kalman filter in the loop to estimate states.

For the case where a Kalman filter is necessary to estimate states, a technique was presented to robustify a controlled system against differences that exist between the controller design model and the real world system. The technique was developed for a continuous-time system and several extensions to discrete-time systems were presented.

A disadvantage of the type of controller considered in this chapter is that it will not regulate the states to zero in the face of unmodeled disturbances. Also, it only regulates the deviations in the states from an equilibrium position and does not allow the system to track a desired piecewise constant non-zero input. Chpater III presents the development for a type of controller that does exhibit these characteristics, based on an LQG methodology.

III. PI CONTROLLERS

3.1 Introduction

This chapter presents the equations used for designing Proportional-plus-Integral (PI) controllers for a system modeled as linear, time-invariant, and driven by stationary white Gaussian noise and deterministic inputs. The model used for this thesis is described in Chapter V.

The advantage of a PI controller over the regulator described in Chpater II is that, in the absence of stochastic inputs, it will maintain the system output at some nonzero commanded value with zero steadystate error, even in the presence of unmodeled constant disturbances. This is known as a "Type-1" property (Ref 4). This property is achieved by integration of the error in the controlled variables, i.e., integration of the difference between achieved and desired values of the controlled variables. In discrete-time feedback control, the PI effect is achieved by performing summation (or pseudo-integration, often interpreted as Euler integration) on either the difference between the actual and desired system outputs (regulation error). In this thesis, this structure is obtained via LQG synthesis applied to an augmented system composed of the original system states and the controls (Ref 24:141-150). Two possible implementations are "position form" and "incremental form". In position form, the control $\underline{u}(t_i)$ is specified in terms of the current position of the system state, $\underline{x}(t_i)$, as in the LQ regulator solution of the previous chapter. In incremental form, only changes in states and commands from the previous values are used to generate increments in control relative to the value at the previous sample time. For implemention, an incremental form PI controller is generally preferable because it is not necessary to provide initial values for the controller states as in the position form. Also it lends iteself more readily to relinearizations about new nominal values and to anti-windup compensation (Ref 24).

The optimal deterministic (LQ) PI controller is developed in the first section of this chapter for a sampled-data system. Next, the Doyle and Stein technique is applied to a PI controller. Then the controller is put into the generic format presented in Chapter II, with a Kalman filter in the loop to provide estimates of the states to the controller. Once the generic form is available, the performance of the controller may be evaluated using the method of the previous chapter.

The software developed to design optimal PI controllers is described in Reference 13. Modifications to the software are described in References 26, 27 and 29. The performance analysis software is described in Reference 29. The above references deal primarily with the design of controllers that employ a PI regulator in the inner feedback loop and a Command Generator Tracker (CGT) to provide inputs to the system (Ref 12). This chapter contains the development of only the inner loop PI regulator.

3.2 Sampled-Data PI controller

The following development for a PI controller, based on augmentation to the original system state relations of equations for pseudo-integration of the control input rates, is found in Reference 24:Ch 14.

3.2.1 Control-Rate Pseudo Integration

Recall from Chapter II that an equivalent discrete-time difference equation can be generated for the continuous-time system model in the form of

$$\underline{x}(t_{i+1}) = \phi\underline{x}(t_i) + B_d\,\underline{u}(t_i) \qquad (3\text{-}1)$$

where, by the certainty equivalence principle (Ref 24:Ch 13), the noise vector was deleted.

Define the perturbation state and control variables to be

$$\delta\underline{x}(t_i) = \underline{x}(t_i) - \underline{x}_o \qquad (3\text{-}2a)$$

$$\delta\underline{u}(t_i) = \underline{u}(t_i) - \underline{u}_o \qquad (3\text{-}2b)$$

where $\underline{x}_o$ and $\underline{u}_o$ are the nominal values of the states and the controls to maintain the system at its equilibrium "trim" operating condition such that controlled variables assume the desired values. If an equation for the output of the system is given by

$$\underline{y}(t_i) = C\underline{x}(t_i) + D_y\,\underline{u}(t_i) \qquad (3\text{-}3)$$

then the nominal control, $\underline{u}_o$, to hold the system at that equilibrium operating point is found as the solution to

$$\underline{x}_o = \phi\underline{x}_o + B_d\,\underline{u}_o \qquad (3\text{-}4a)$$

$$\underline{y}_d = C\underline{x}_o + D_y\,\underline{u}_o \qquad (3\text{-}4b)$$

or

$$\begin{bmatrix} \underline{0} \\ \underline{y}_d \end{bmatrix} = \begin{bmatrix} (\phi-I) & B_d \\ C & D_y \end{bmatrix} \begin{bmatrix} \underline{x}_o \\ \underline{u}_o \end{bmatrix} \qquad (3\text{-}5)$$

where $\underline{y}_d$ is the desired system output. Then, (3-5) can be solved as

$$\begin{bmatrix} \underline{x}_o \\ \underline{u}_o \end{bmatrix} = \begin{bmatrix} (\phi - I) & B_d \\ C & D_y \end{bmatrix}^{-1} \begin{bmatrix} \underline{0} \\ \underline{y}_d \end{bmatrix} = \begin{bmatrix} \pi_{11} & \pi_{12} \\ \pi_{21} & \pi_{22} \end{bmatrix} \begin{bmatrix} \underline{0} \\ \underline{y}_d \end{bmatrix} \qquad (3\text{-}6)$$

or

$$\underline{x}_o = \pi_{12}\, \underline{y}_d \qquad \underline{u}_o = \pi_{22}\, \underline{y}_d \qquad (3\text{-}7)$$

The difference in perturbation control variable is

$$\delta\underline{u}(t_{i+1}) - \delta\underline{u}(t_i) = [\underline{u}(t_{i+1}) - \underline{u}_o] - [\underline{u}(t_i) - \underline{u}_o] \qquad (3\text{-}8)$$

or, equivalently

$$\delta\underline{u}(t_{i+1}) = \delta\underline{u}(t_i) + [\underline{u}(t_{i+1}) - \underline{u}(t_i)] \qquad (3\text{-}9)$$

The above equation has the form of an update relation for $\delta\underline{u}(t_{i+1})$. If the second term is thought of as an Euler integration of the time rate-of-change of the control input $\underline{u}$ at time t_i, then the change in $\underline{u}$, i.e., the bracketed term in Equation (3-9), can be written as

$$\Delta\underline{u}(t_i) = \Big[\underline{u}(t_{i+1}) - \underline{u}(t_i)\Big] = \dot{\underline{u}}(t_i)\Delta t \qquad (3\text{-}10)$$

where Δt is the sample time of the controller. Thus, $\Delta\underline{u}(t_i)$ is termed the control pseudo-rate and Equation (3-9) becomes

$$\delta\underline{u}(t_{i+1}) = \delta\underline{u}(t_i) + \Delta\underline{u}(t_i) \qquad (3\text{-}11)$$

At this point, an augmented state description can be written using Equations (3-1) and (3-11) for the perturbation variables $\delta\underline{x}(t_i)$ and $\delta\underline{u}(t_i)$:

$$\begin{bmatrix} \delta\underline{x}(t_{i+1}) \\ \\ \delta\underline{u}(t_{i+1}) \end{bmatrix} = \begin{bmatrix} \Phi & B_d \\ \\ 0 & I \end{bmatrix} \begin{bmatrix} \delta\underline{x}(t_i) \\ \\ \delta\underline{u}(t_i) \end{bmatrix} + \begin{bmatrix} 0 \\ \\ I \end{bmatrix} \Delta\underline{u}(t_i) \quad (3\text{-}12)$$

The control pseudo-rate is now considered to be the input to this augmented perturbation state equation.

3.2.2 Optimal Regulator Solution

The optimal regulator solution described in Section 2.5 can now be applied to the augmented system above, subject to a quadratic cost criterion

$$J_c = \frac{1}{2} \begin{bmatrix} \delta\underline{x}(t_{N+1}) \\ \\ \delta\underline{u}(t_{N+1}) \end{bmatrix}^T \begin{bmatrix} X_f & 0 \\ \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \delta\underline{x}(t_{N+1}) \\ \\ \delta\underline{u}(t_{N+1}) \end{bmatrix}$$

$$+ \frac{1}{2} \sum_{i=-1}^{N} \begin{bmatrix} \delta\underline{x}(t_i) \\ \delta\underline{u}(t_i) \\ \Delta\underline{u}(t_i) \end{bmatrix}^T \begin{bmatrix} X_{11} & X_{12} & S_1 \\ X_{12}^T & X_{22} & S_2 \\ S_1^T & S_2^T & U \end{bmatrix} \begin{bmatrix} \delta\underline{x}(t_i) \\ \delta\underline{u}(t_i) \\ \Delta\underline{u}(t_i) \end{bmatrix} \quad (3\text{-}13)$$

where X_{11} places a weighting on state deviations away from the nominal $\underline{x}_o$, X_{22} places a weighting on control deviations away from $\underline{u}_o$, and U places a weighting on the control pseudo-rates, $\Delta\underline{u}(t_i)$. Notice that this term allows the designer to place a weighting on the rate of change of control inputs, i.e., to guard against commanding actuator rates that are beyond the physical limits of the actuators.

The term X_f applies a weighting on deviations of the states at the final time. The cross-terms X_{12}, S_1 and S_2 arise as the continuous-time cost is converted to a discrete-time cost, as demonstrated in Reference 24 and 12. In Equation (3-13), notice that the index for the summation begins at -1. This places a weighting on the potentially large control differences which may occur at the initial time due to a change in the setpoint (Ref 24:142).

To apply the optimal regulator solution to Equations (3-12) and (3-13), redefine the matrices as follows

$$\phi_\delta = \begin{bmatrix} \phi & B_d \\ 0 & I \end{bmatrix} \tag{3-14}$$

$$B_d = \begin{bmatrix} 0 \\ I \end{bmatrix} \tag{3-15}$$

$$\delta\underline{x}_\delta = \begin{bmatrix} \delta\underline{x} \\ \delta\underline{u} \end{bmatrix} \tag{3-16}$$

$$X_{f\delta} = \begin{bmatrix} X_f & 0 \\ 0 & 0 \end{bmatrix} \tag{3-17}$$

$$X_\delta = \begin{bmatrix} X_{11} & X_{12} \\ X_{12}^T & X_{22} \end{bmatrix} \tag{3-18}$$

$$S_\delta = \begin{bmatrix} S_1 \\ S_2 \end{bmatrix}$$

Thus, the state equation and cost function become

$$\delta\underline{x}_\delta\ (t_{i+1}) = \phi_\delta\, \delta\underline{x}_\delta(t_i) + B_\delta\ \ \Delta\underline{u}(t_i) \tag{3-20}$$

$$J_c = \frac{1}{2}\ \ \delta\underline{x}_\delta^T(t_{N+1})\ \ X_{f\delta}\ \ \delta\underline{x}_\delta(t_{N+1})$$

$$+ \frac{1}{2}\ \sum_{i=-1}^{N} \begin{bmatrix} \delta\underline{x}_\delta(t_i) \\ \Delta\underline{u}(t_i) \end{bmatrix}^T \begin{bmatrix} X_\delta & S_\delta \\ S_\delta^T & U \end{bmatrix} \begin{bmatrix} \delta\underline{x}_\delta(t_i) \\ \Delta\underline{u}(t_i) \end{bmatrix} \tag{3-21}$$

Then, the optimal, constant-gain LQ regulator solution is given by

$$\Delta\underline{u}(t_i) = -G_c^*\ \delta\underline{x}_\delta(t_i) \tag{3-22}$$

where

$$G_c^* = [G_{c_1}^* \quad G_{c_2}^*] = [U + B_\delta^T\, \bar{K}_c\ B_\delta]^{-1}\ [B_\delta^T\, \bar{K}_c\ \phi_\delta + S_\delta^T] \tag{3-23}$$

and $\bar{K}_c$ is the steady-state solution to the backward Riccati difference equation

$$K_c(t_i) = X_\delta + \phi_\delta^T K_c(t_{i+1}) [\phi_\delta - B_\delta G_c^*] - S_\delta G_c^* \qquad (3\text{-}24)$$

solved backwards from the terminal condition

$$K_c(t_{N+1}) = X_{f\delta} \qquad (3\text{-}25)$$

The optimal control input is given by Equation (3-22) in partitioned form as

$$\Delta \underline{u}^*(t_i) = - G_{c_1}^* \delta\underline{x}(t_i) - G_{c_2}^* \delta\underline{u}(t_i) \qquad (3\text{-}26)$$

Combining this with Equation (3-11) yields

$$\delta\underline{u}^*(t_{i+1}) = \delta\underline{u}^*(t_i) - [G_{c_1}^* \quad G_{c_2}^*] \begin{bmatrix} \delta\underline{x}(t_i) \\ \\ \delta u^*(t_i) \end{bmatrix} \qquad (3\text{-}27)$$

Unfortunately, the above control law does not achieve the desired Type 1 property.

3.3 Achieving Type-1 Control

The desired integral characteristics for the controller can be achieved by manipulation into the form of a continuous-time PI controller (Ref 1)

$$\underline{u}^*(t) = -K_x\underline{x}(t) + K_z \int_{t_o}^{t} [\underline{y}_d - \underline{y}(\tau)]\, d\tau \qquad (3\text{-}28)$$

where $\underline{y}_d$ is the desired output and $\underline{y}(t)$ is the actual system output defined by

$$\underline{y}(t) = C\underline{x}(t) + D_y\underline{u}(t) \qquad (3\text{-}29)$$

A discrete-time equivalent is given by

$$\underline{u}^*(t_i) = -K_x\underline{x}(t_i) + K_z \sum_{j=-1}^{i-1} [\underline{y}_d - \underline{y}(t_j)] \qquad (3\text{-}30)$$

In incremental form, the above equation is expressed as

$$\underline{u}^*(t_{i+1}) = \underline{u}^*(t_i) - K_x[\underline{x}(t_{i+1}) - \underline{x}(t_i)] + K_z[\underline{y}_d - \underline{y}(t_i)] \quad (3\text{-}31)$$

Define a perturbation output equation as

$$\delta\underline{y}(t_i) = \underline{y}(t_i) - \underline{y}_d = C\delta\underline{x}(t_i) + D_y\delta\underline{u}(t_i) \qquad (3\text{-}32)$$

and the control law as

$$\delta\underline{u}^*(t_{i+1}) = \delta\underline{u}^*(t_i) - K_x[\delta\underline{x}(t_{i+1}) - \delta\underline{x}(t_i)]$$

$$- K_z\ [C\delta\underline{x}(t_i) + D_y\delta\underline{u}(t_i)] \qquad (3\text{-}33)$$

The above equation can be rewritten using Equation (3-9) as

$$\delta\underline{u}^*(t_{i+1}) = \delta\underline{u}^*(t_i) - [K_x \quad K_z] \begin{bmatrix} (\phi-I) & B_d \\ C & D_y \end{bmatrix} \begin{bmatrix} \delta\underline{x}(t_i) \\ \delta\underline{u}(t_i) \end{bmatrix} \qquad (3\text{-}34)$$

Equating Equations (3-34) and (3-27), it is seen that

$$[K_x \quad K_z] = [G_{c_1}{}^* \quad G_{c_2}{}^*] \begin{bmatrix} (\phi-I) & B_d \\ C & D_y \end{bmatrix}^{-1} \qquad (3\text{-}35)$$

$$\begin{bmatrix} K_x & K_z \end{bmatrix} = [G_{c_1}^{*} \quad G_{c_2}^{*}] \begin{bmatrix} \pi_{11} & \pi_{12} \\ & \\ \pi_{21} & \pi_{22} \end{bmatrix} \tag{3-36}$$

The values for the feedback gains K_x and K_z are thus defined by

$$K_x = G_{c_1}^{*} \; \pi_{11} + G_{c_2}^{*} \; \pi_{21} \tag{3-37a}$$

$$K_z = G_{c_1}^{*} \; \pi_{12} + G_{c_2}^{*} \; \pi_{22} \tag{3-37b}$$

once $G_{c_1}^{*}$ and $G_{c_2}^{*}$ are available from the augmented state regulator solution and π_{11}, π_{12}, π_{21}, and π_{22} are obtained from Equation (3-35) and (3-36). The final form for the PI cotroller which achieves Type-1 control, implemented in incremental form is

$$\delta\underline{u}^{*}(t_{i+1}) = \delta\underline{u}^{*}(t_i) - K_x[\delta\underline{x}(t_{i+1}) - \delta\underline{x}(t_i)] \tag{3-38}$$

$$+ K_z\ [\underline{y}_d(t_{i+1}) - \underline{y}(t_i)] \tag{3-38}$$

Notice that the different time arguments on the $\underline{y}$ terms are correct (Ref 24:147).

3.4 Doyle and Stein Technique for PI Controllers

The technique of injecting white input noise into a controlled system is accomplished for a sampled-data PI controller in a manner similar to that described in Section 2.7.2 for a sampled-data LQ regulator.

The equivalent discrete-time, stochastic difference equation to describe the system to be controlled is given by

$$\underline{x}(t_{i+1}) = \phi\underline{x}(t_i) + B_d\,\underline{u}(t_i) + G_d\,\underline{w}_d(t_i) \qquad (3\text{-}39)$$

where the noise vector has now been included. The discrete-time input vector, B_d, and the noise covariance, Q_d, are formed by solving the following equations

$$B_d = \int_{t_i}^{t_{i+1}} \phi(t_{i+1}-\tau)\;B\;d\tau \qquad (3\text{-}40)$$

$$Q_d = \int_{t_i}^{t_{i+1}} \phi(t_{i+1}-\tau)\;GQG^T\;\phi^T(t_{i+1}-\tau)\;d\tau \qquad (3\text{-}41)$$

where $Q_d = E\{\underline{w}_d(t_i)\underline{w}_d^T(t_i)\}$. G_d is now defined to be the identify matrix, I.

Recall from Chapter II that input noise was added to the system by modifying the Q_d matrix via

$$Q_d(q) = Q_{do} + q^2BVB^T\Delta t \qquad (3\text{-}42)$$

where Q_{do} is now the original noise covariance given in Equation (3-40), B is the input vector from the continuous-time state differential equation, q^2 is a scalar design parameter which adjusts the strength of the input noise, and Δt is the sample time of the discrete controller. This, in effect, makes a first-order approximation to the discrete-time input noise.

However, the discrete-time input vector is available from Equation (3-40) and is approximated to first order by $B_d = B\Delta t$, and so it will be used to modify the Q_d matrix for PI controllers. The resulting modified noise covariance matrix is

$$Q_d(q) = Q_{do} + q^2 \, B_d \, V \, B_d^T \, /\Delta t \tag{3-43}$$

where V is a nonsingular matrix, chosen to be the identity matrix for this thesis. Notice that the Kalman filter design is the same regardless of the type of controller implemented so that robustification would be accomplished in the same manner as described in Chapter II. The structure of the sampled-data PI controller is shown in Figure (3-1). The Kalman filter of Figure (3-1) is the same as for the LQG regulator shown in Figure (2-4).

3.5 Performance Analysis for PI Controllers

The performance analysis for a PI controller is accomplished by following the same procedure given in Section 2.6 for LQ regulators. As given in Equation (3-38), the incremental form for the optimal, deterministic PI control law is given by

$$\begin{aligned} \delta\underline{u}^*(t_{i+1}) = \delta\underline{u}^*(t_i) - K_x \, [\delta\underline{x}(t_{i+1}) - \delta\underline{x}(t_i)] \\ + K_z \, [\underline{y}_d(t_{i+1}) - \underline{y}(t_i)] \end{aligned} \tag{3-44}$$

Again, the differing time arguments on the $\underline{y}$ terms are correct.

When a Kalman filter is employed to estimate states for the controller, the conditional mean and covariance of the states are propagated between sample times by Equations (2-76a) and (2-76b). At the sample times, when measurements become available, the conditional mean and

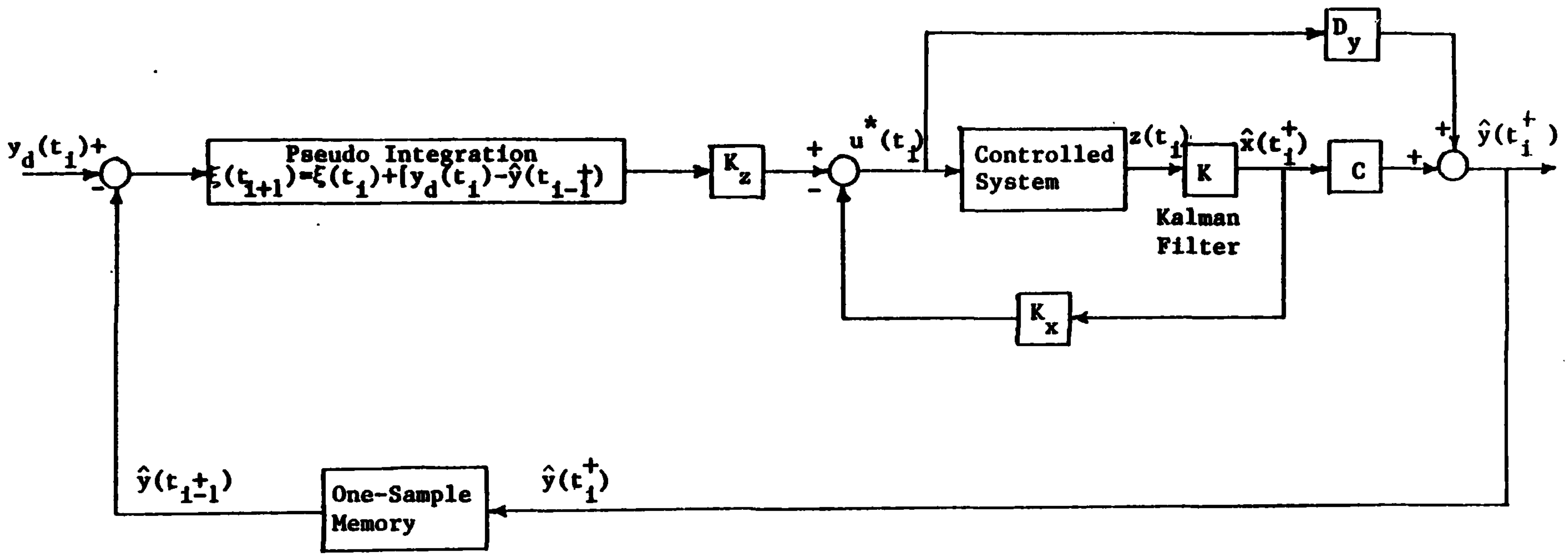


Figure 3-1: Sampled-Data PI Controller

covariance are updated by Equations (2-77) and (2-78), where the Kalman filter gain is given in Equation (2-79). Using the control law of Equation (3-44) and the Kalman filter equations, the PI controller can be put into the generic format introduced in Equation (2-82) and (2-83):

$$\underline{u}^{*}(t_{i+1}) + G_{cx}\,\underline{x}_c(t_i) + G_{cz}\,\underline{z}(t_i) + G_{cy}\,\underline{y}_d(t_i) \tag{3-45a}$$

$$\underline{x}_c(t_{i+1}) = \phi_c\,\underline{x}_c(t_i) + B_{cz}\,\underline{z}(t_i) + B_{cy}\,\underline{y}_d(t_i) \tag{3-45b}$$

where the subscript c refers to states of the controller. The gain matrices of Equations (3-45a) and (3-45b) are evaluated explicitly in Appendix A.

At this point, the performance of the PI controller may be evaluated using the method of Section 2.6, evaluating the statistical characteristics of an augmented vector

$$\underline{y}_a(t_i) = \begin{bmatrix} \underline{x}_t(t_i) \\ \\ \underline{u}(t_i) \end{bmatrix} \tag{3-46}$$

which is composed of the states of the truth model and the controls generated by the PI controller.

3.6 Summary

The chapter has presented the equations for designing and evaluating PI controllers based on an LQG methodology. The advantage of a PI controller over the LQG regulator discussed in Chapter II is that it will track a desired output with zero steady-state error (in a deterministic setting, or in zero steady-state mean error in a stochastic environment) even in the face of unmodeled constant disturbance. The type chosen for implementation was an incremental form of PI controller that is based on augmentation of system state equations with relations involving pseudointegration of the control rates.

The Doyle and Stein stability robustness enhancement technique of inputting white noise into the system model at the control entry points during filter tuning was applied to PI controllers. The following chapter will extend this method to allow time-correlated input noise for the case where robustification is desired over only a limited frequency range.

Chapter IV. Time-Correlated Input Noise

4.1 Introduction

Section 2.4 introduced the idea of robustifying a Kalman filter by adding pseudonoise to a system at the point of entry of the control inputs, $\underline{u}$. By increasing the intensity of this pseudonoise, the stability robustness properties of a full-state feedback system could by asymptotically recovered by the observer-based sytem.

One disadvantage of this techniques is that the noise addition degrades the performance of the system at the design conditions. The white noise adds uncertainty to the model with the same intensity throughout the frequency spectrum. It may be desired to robustify the Kalman filter only over a certain frequency range where confidence in the adequacy of the controller design model may be low. Thus, a natural extension to the Doyle and Stein technique (Ref 6) is to add time-correlated (colored) input noise where the strength of the noise is highest for the frequency range of interest. This chapter develops this idea and describes how to apply it to the types of controllers discussed in Chapter II and III (Ref 15;16;18).

In the first section, a stochastic model is developed for the fictitious colored noise process. Then, this is augmented with the model used to describe the system. Finally, the colored noise processes to be used in this thesis are described, including the shaping filters to generate them.

4.2 Stochastic Model

Stationary, time-correlated input noise is represented as the steady-state output of a linear time-invariant system driven by white Gaussian noise as

$$\dot{\underline{x}}_u(t) = F_u\, \underline{x}_u(t) + G_u\, \underline{w}_u(t) \tag{4-1a}$$

$$\underline{n}_u(t) = H_u\, \underline{x}_u(t) + D_u\, \underline{w}_u(t) \tag{4-1b}$$

The subscript u refers to uncertainty parameters. The statistics of the stationary white noise, $\underline{w}_u(t)$, are given by

$$E\{\underline{w}_u(t)\} = \underline{0} \tag{4-2a}$$

$$E\{\underline{w}_u(t)\, \underline{w}_u^T(t+\tau)\} = Q_u \delta(\tau) \tag{4-2b}$$

The colored noise process, $\underline{n}_u(t)$, is added to the system model, given by

$$\dot{\underline{x}}(t) = F\, \underline{x}(t) + B\, \underline{u}(t) + G\, \underline{w}(t) \tag{4-3a}$$

$$\underline{z}(t) = H\, \underline{x}(t) + \underline{v}(t) \tag{4-3b}$$

at the point of entry of $\underline{u}$. This results in

$$\dot{\underline{x}}(t) = F\, \underline{x}(t) + B\left[\underline{u}(t) + \underline{n}_u(t)\right] + G\, \underline{w}_u(t) \tag{4-4}$$

Define an augmented vector to be the system states and the uncertainty model states

$$\underline{x}_a(t) = \begin{bmatrix} \underline{x}(t) \\ \\ \underline{x}_u(t) \end{bmatrix} \tag{4-5}$$

Next, augment the uncertainty state differential equations with those of the original system. This yields an augmented system equation, given by

$$\dot{\underline{x}}_s(t) = F_s\, \underline{x}_s(t) + B_s\, \underline{u}(t) + G_s\, \underline{w}_s(t) \tag{4-6a}$$

$$\underline{z}_s(t) = H_s\, \underline{x}_s(t) + \underline{v}(t) \tag{4-6b}$$

The above matrices are described by

$$F_s = \begin{bmatrix} F & BH_u \\ 0 & F_u \end{bmatrix} \tag{4-7}$$

$$B_s = \begin{bmatrix} B \\ 0 \end{bmatrix} \tag{4-8}$$

$$G_s = \begin{bmatrix} G & BD_u \\ 0 & G_u \end{bmatrix} \tag{4-9}$$

$$H_s = \begin{bmatrix} H & 0 \end{bmatrix} \tag{4-10}$$

The noise vector, $\underline{w}_s(t)$, is given by

$$\underline{w}_s(t) = \begin{bmatrix} \underline{w}(t) \\ \underline{w}_u(t) \end{bmatrix} \tag{4-11}$$

with covariance kernal $E\{\underline{w}_s(t)\underline{w}_s^T(t+\tau)\} = Q_s\delta(\tau)$, where

$$Q_s = \begin{bmatrix} Q & 0 \\ 0 & Q_u \end{bmatrix} \qquad (4\text{-}12)$$

The augmented system model described in Equations (4-5) through (4-12) is used to design either the continuous-time Kalman filter in Equations (2-12a) and (2-12b) or the sampled-data Kalman filter in Equations (2-76) through (2-79). However, it is not necessary to feed back the uncertainty model states in the controller algorithm; they are used only to modify the filter within the overall controller. Therefore, the original system model is used to design the LQ regulator of Chapter II or the PI controller of Chapter III. However, the dimensions of the Kalman filter and the controller are now incompatible. This is solved by augmenting the controller gain matrix with zeros

$$G_{c_s}^* = \left[\, G_c^* \;\vdots\; 0 \,\right] \qquad (4\text{-}13)$$

Thus, the order of the original system has been increased to reflect the adding of uncertainty model states, but these are fed back through the controller with zero gains.

At this point, the control law is put into the continuous-time generic form of Equations (2-25) and (2-26) or the sampled-data form of Equations (2-82) and (2-83). then, the performance of the robustified system is evaluated using the performance analysis described in Chapter II.

4.3 Shaping Filter Design

The time-correlated noise processes to be examined in this thesis are generated by inputting white noise to first- or second-order shaping filters as described in Reference 23:180-186. The following sections describe the form of the shaping filters.

4.3.1 First-Order Colored Noise Processes

A noise process which has low intensity at low frequencies and high intensity at high frequencies (or high intensity at low frequencies and low intensity at high frequencies) can be generated as the output of a system with a transfer function of the form

$$\frac{x_u(s)}{w_u(s)} = \frac{s + a}{s + b} \tag{4-14}$$

In Chapter V, it is shown that the model used for this thesis is adequate for low frequencies but may not be for higher frequencies. Thus, for this case, the strength of the noise should be kept low at low frequencies and then increased where the adequacy of the model is in doubt. Figure (4-1) shows the power spectral density function for this process.

Using the standard controllable form given in Reference 23:Ch 2, the state-space representation for Equation (4-14) is

$$\dot{x}_u(t) = -b\, x_u(t) + w_u(t) \tag{4-15a}$$

$$n_u(t) = (a - b)\, x_u(t) + w_u(t) \tag{4-15b}$$

For this first-order shaping filter, the uncertainty model matrices are thus scalar and given by

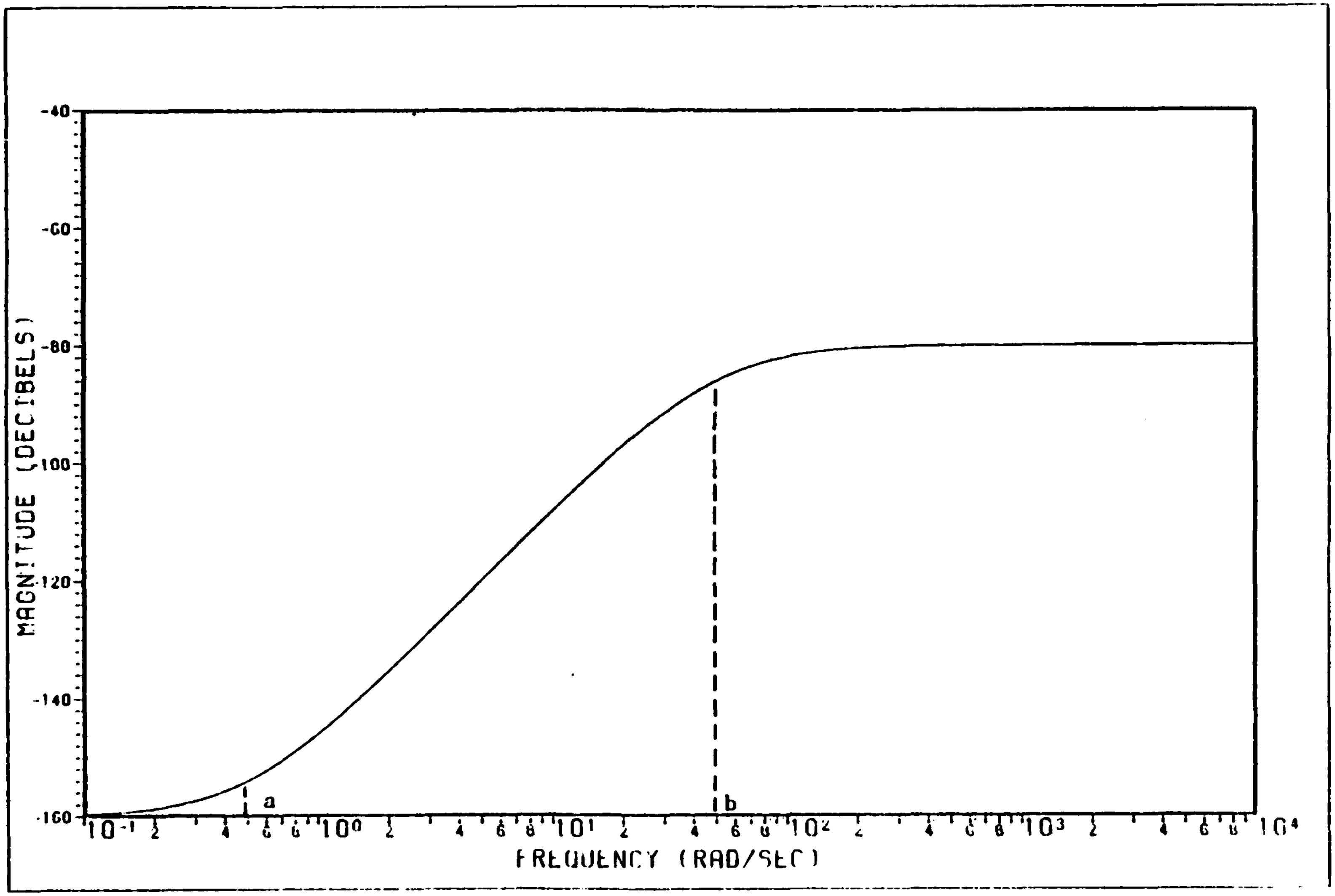


Figure 4-1: Power Spectral Density Function for a First-Order Shaping Filter

$$F_u = -b \tag{4-16a}$$

$$B_u = 1 \tag{4-16b}$$

$$H_u = (a - b) \tag{4-16c}$$

$$D_u = 1 \tag{4-16d}$$

Note that a noise process is generated for each control applied to the system. Thus, the dimension of the augmented system is that of the basic system plus the number of controls.

4.3.2 Second-Order Colored Noise Processes

A noise process which has high intensity over only a limited frequency range can be generated by passing white noise through a linear system model that has a transfer function of the form

$$\frac{x_u(s)}{w_u(s)} = \frac{s + c}{(s + d)\,(s + e)} \tag{4-17}$$

If it is the case where the adequacy of the design model is in doubt over a limited frequency range, then a second-order shaping filter as in Equation (4-17) would be appropriate. The power spectral density function for this process is shown in Figure (4-2).

A state-space representation for this process, using standard controllable form is given by

$$\begin{bmatrix} \dot{x}_{u_1}(t) \\ \dot{x}_{u_2}(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -de & -(d + e) \end{bmatrix} \begin{bmatrix} x_{u_1}(t) \\ x_{u_2}(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} w_u(t) \tag{4-18a}$$

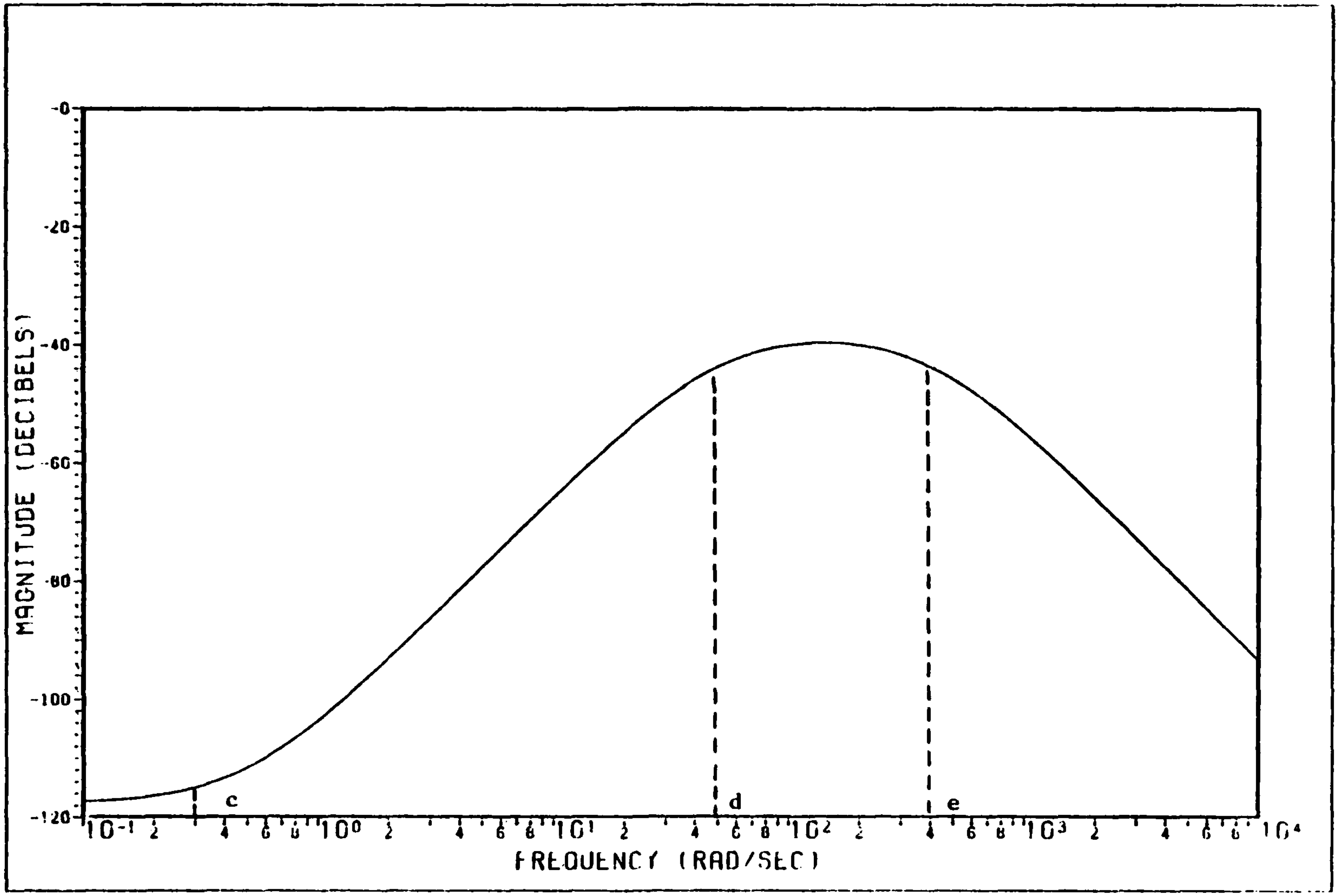


Figure 4-2: Power Spectral Density Function for a Second-Order Colored Noise Process

$$n_u(t) = \begin{bmatrix} c & 1 \end{bmatrix} \begin{bmatrix} x_{u_1}(t) \\ x_{u_2}(t) \end{bmatrix} \tag{4-18b}$$

For this second-order shaping filter, the uncertainty model matrices are thus given by

$$F_u = \begin{bmatrix} 0 & 1 \\ -de & -(d + e) \end{bmatrix} \tag{4-19a}$$

$$B_u = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{4-19b}$$

$$H_u = \begin{bmatrix} c & 1 \end{bmatrix} \tag{4-19c}$$

$$D_u = \begin{bmatrix} 0 \end{bmatrix} \tag{4-19d}$$

For this case, the dimensionality of the original system is increased by two times the number of controls when these shaping filter states are augmented to the system states.

The above technique for robustifying the Kalman filter in a control system is incorporated into two subroutines added to the LQG regulator design program described in Reference 21. Source code and instructions for running the program with the additional routines are listed in Appendix B.

Similarly, for PI controllers, the user must input the original design model, before augmenting the shaping filter states, to the program described in Reference 13. The original model is used to design the PI controller. Then, the shaping filter states are augmented with the model for the Kalman filter design. Finally, the PI controller feedback gain matrix is augmented with zeros to make the dimensions of the two solutions compatible. Modifications to the software in Reference 13 are listed in Appendix C.

4.4 Summary

This chapter has presented an extension to the Doyle and Stein robustness method of injecting white input noise into a controlled system during the process of tuning the Kalman filter. The extension allows time-correlated noise to be injected into the system model, which is accomplished by augmenting shaping filter states to the state differential equations of the system to be controlled. Thus, the strength of the colored noise can be concentrated at frequencies where robustification is desired and attenuated elsewhere.

Two types of shaping filters to generate the colored noise were examined. The first concentrated the strength of the noise at higher frequencies. The second concentrated the strength of the noise over a limited frequency range. Then, the method was applied to the types of controllers discussed in this thesis.

At this point, the types of controllers which are designed for this thesis have been discussed. In addition, techniques to enhance the robustness characteristics of the controllers have been presented. The following chapter presents the linear, time-invariant model to which the above mentioned methods are applied in this thesis.

Chapter V. AFTI/F-16 Flight Control Design

5.1 Introduction

This chapter presents the longitudinal dynamic equations for the Advanced Fighter Technology Integration aircraft, an F-16 aircraft (AFTI/F-16) modified for advanced controls research. For longitudinal control, the aircraft is equipped with a decoupled horizontal tail and trailing edge flap which make it possible for the AFTI/F-16 to perform such maneuvers as pitch pointing. A pitch-pointing maneuver allows the aircraft's attitude angle to be changed while holding the flight path angle constant. The controllers to be designed lend themselves readily to multiple-input multiple-output (MIMO) system methods such as those described in this thesis. The aircraft equations are adequately portrayed as linear and time invariant by linearization about specified trim conditions in the aircraft's flight envelope. The controllers described in earlier chapters will be designed for the AFTI/F-16, applying the techniques of tuning the Kalman filter for robustness purposes by injecting fictitious white or time-correlated noise into the system model at the point of entry of the control inputs.

Data used to form the AFTI/F-16 longitudinal equations was taken from References 11 and 12. Reference 11 lists dimensional stability derivatives in a body-axis coordinate system for the aircraft, but does not include data for the trailing-edge flap. This information is given in Reference 12 in a stability-axis coordinate system. The necessary transformations were performed to convert all values to the body-axis frame.

The development of a linear perturbation model for an aircraft's longitudinal dynamics is found in References 7;8;9;10;25. The model for clear air turbulence is developed in Reference 17 and used in Reference 12. Both these models are incorporated to form the AFTI/F-16 equations described later in this chapter.

The first section of this chapter describes a thirteen-state linear perturbation truth model for the AFTI/F-16. This thirteen-state model is unobservable because a measurement is not available for all four aircraft states. Therefore, the unobservable state is deleted from the full model to leave a twelve-state truth model. This model, described in a second section, will be used to evaluate the performance of lower order controller models. The lower order (eight-state) controller model is described in a third section. The final section gives the data and longitudinal equations for one twelve-state truth model at an off-design flight condition. This will be used to evaluate the robustness of the controllers designed.

5.2 AFTI/F-16 Truth Model

The longitudinal aircraft truth model consists of four aircraft states, three turbulence (gust) states, and six additional states to describe the third order dynamics for each of the horizontal tail and trailing-edge-flap actuators. The aircraft states are: Θ, attitude angle; α, angle of attack, q, pitch rate, and; u, forward perturbation velocity. There are two angle of attack gust states and one pitch-rate gust state. Noise-corrupted measurements are available for Θ, α, and q, and the measurement for α includes an angle-of-attack gust state. The actuator and gust models will be described first.

5.2.1 Actuator Dynamics

The actuators for the horizontal tail and trailing-edge-flap are modeled as third-order systems, yielding delivered angle δ in response to commanded angle δ_{com}, of the form

$$\frac{\delta(s)}{\delta_{com}(s)} = \frac{(1/T_s)\,(\omega_n^{\,2})}{(s + 1/T_s)\,(s^2 + 2\xi\,\omega_n s + \omega_n^{\,2})} \tag{5-1}$$

Multiplying the numerator and denominator terms yields

$$\frac{\delta(s)}{\delta_{com}(s)} = \frac{a_o}{s^3 + a_2 s^2 + a_1 s + a_o} \tag{5-2}$$

Using the standard observable state space representation given in Reference 23:Ch 2, the corresponding model is

$$\begin{bmatrix} \dot{\delta}(t) \\ \dot{\delta}_1(t) \\ \dot{\delta}_2(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -a_o & -a_1 & -a_2 \end{bmatrix} \begin{bmatrix} \delta(t) \\ \delta_1(t) \\ \delta_2(t) \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \delta_{com}(t) \tag{5-3a}$$

$$\delta(t) = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \delta(t) \\ \delta_1(t) \\ \delta_2(t) \end{bmatrix} \tag{5-3b}$$

where the b_i's arise when taking the Laurent series for G(s). Notice that δ_{com} is the command surface deflection and δ is the surface deflection output from the actuators. Two intermediate states, δ_1 and δ_2, are needed to describe fully the actuator dynamics.

The values chosen for the parameters in Equation (5-1) are specified in Reference 27 as

$$\frac{\delta(s)}{\delta_{com}(s)} = \frac{(20.2)\ (71.4)^2}{(s + 20.2)\ (s^2 + 2(.736)\ (71.4) + 71.4^2)} \tag{5-4}$$

The same actuator equations are used for both the horizontal tail and trailing-edge-flap. Expanding Equation (5-4) yields

$$\frac{\delta(s)}{\delta_{com}(s)} = \frac{102980}{(s^3 + 125.3s^2 + 7221s + 102980)} \tag{5-5}$$

Thus the resulting state equation is

$$\begin{bmatrix} \dot{\delta}(t) \\ \dot{\delta}_1(t) \\ \dot{\delta}_2(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -102980. & -7221. & -125.3 \end{bmatrix} \begin{bmatrix} \delta(t) \\ \delta_1(t) \\ \delta_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 102980 \end{bmatrix} \delta_{com}(t) \tag{5-6}$$

5.2.2 Turbalence State Equations

The model used for clear air turbulence is found in References 12 and 17. The state equations for the three gust states are given by

$$\dot{\alpha}_g' = a_{g_1}' \alpha_g + n_\alpha \tag{5-7a}$$

$$\dot{\alpha}_g' = a_{g_2} \alpha_g' + a_g \alpha_g = a_n n_\alpha \tag{5-7b}$$

$$\dot{q}_g = a_{g_d} \dot{\alpha}_g + q_{g_1} q_g \tag{5-7c}$$

The noise source, n_α, is modeled as a white Gaussian noise with statistics

$$E\{n_\alpha(t)\} = 0 \tag{5-8a}$$

$$E\{n_\alpha(t)\ n_\alpha^T\ (t + \tau)\} = \delta(\tau) \tag{5-8b}$$

The coefficients in Equations (5-7) are computed using the following equations

$$a_{g_1}' = \frac{-V_t}{L_w} \tag{5-9a}$$

$$a_{g_2} = \frac{\sigma_w}{L_w}(1. - \sqrt{3.})\sqrt{\frac{V_t}{L_w}} \tag{5-9b}$$

$$a_g = \frac{-V_t}{L_w} \tag{5-9c}$$

$$a_n = \frac{\sigma_w}{V_t}\sqrt{\frac{3V_t}{L_w}} \tag{5-9d}$$

$$a_{g_d} = \frac{\pi V_t}{4b} \tag{5-9e}$$

$$q_{g_1} = \frac{-\pi V_t}{4b} \tag{5-9f}$$

V_t is aircraft velocity and b is the wing span. The quantities σ_w and L_w are given in Reference 3 as functions of altitude, where σ_w is the root mean square intensity of the clear air turbulence in feet per second and L_w is a reference scale length in feet.

5.2.3 System Equations

The thirteen states for the unobservable truth model are listed in Equation (5-10) on the following page.

As demonstrated in Reference 12, it is convenient to define the truth model equations in two stages. First, define the ssytem matrix, F_t' of Equation (2-25), neglecting all $\dot{\alpha}$ stability derivative terms and also the $\dot{\alpha}_g$ term in Equation (5-7c). Then modify the matrix to incorporate the neglected terms. This is done for mathematical convenience as the $\dot{\alpha}$ terms are neglected in a later truth model. Defining the system matrix in two stages does not change the definition of the states; the second stage is merely a more adequate model. The initial matrix is given in Equation (5-11) on the following page.

To incorporate the neglected terms, rows 2,3,4 and 13 are modified with the following equations (using standard double-index array notation $F_t(I,J)$ for the I-th row and J-th column of F_t).

$$F_t(2,J) = F_t'(2,J) \;/(1. - Z_{\dot{\alpha}}) \qquad (5\text{-}12a)$$

$$F_t(3,J) = F_t'(3,J) + M_{\dot{\alpha}} \times F_t(2,J) \qquad (5\text{-}12b)$$

$$F_t(4,J) = F_t'(4,J) + X_{\dot{\alpha}} \times F_t(2,J) \qquad (5\text{-}12c)$$

$$F_t(13,J) = F_t'(13,J) + a_{g_d} \times F_t'(12,J) \qquad (5\text{-}12d)$$

The column index J ranges from 1 to 13.

$$\underline{x}_t = \begin{bmatrix} \theta(t) \\ \alpha(t) \\ q(t) \\ u(t) \\ \delta_{HT}(t) \\ \delta_{HT_1}(t) \\ \delta_{HT_2}(t) \\ \delta_{TEF}(t) \\ \delta_{TEF_1}(t) \\ \delta_{TEF_2}(t) \\ \alpha_g'(t) \\ \alpha_g(t) \\ q_g(t) \end{bmatrix} \tag{5-10}$$

$$F_t' = \left[\begin{array}{cccc|cccccc|ccc}
0. & 0. & 1. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. \\
-g(\sin\alpha)/u_o & Z_\alpha & 1.+Z_q & Z_u & Z\delta_{HT} & 0. & 0. & Z\delta_{TEF} & 0. & 0. & 0. & Z_\alpha & Z_q \\
0. & M_\alpha & M_q & M_u & M\delta_{HT} & 0. & 0. & M\delta_{TEF} & 0. & 0. & 0. & M_\alpha & M_q \\
-g(\cos\alpha_o) & X_\alpha & X_q-w_o & X_u & X\delta_{HT} & 0. & 0. & X\delta_{TEF} & 0. & 0. & 0. & X_\alpha & X_q \\
\hline
0. & 0. & 0. & 0. & 0. & 1. & 0. & 0. & 0. & 0. & 0. & 0. & 0. \\
0. & 0. & 0. & 0. & 0. & 0. & 1. & 0. & 0. & 0. & 0. & 0. & 0. \\
0. & 0. & 0. & 0. & -a_o & -a_1 & -a_2 & 0. & 0. & 0. & 0. & 0. & 0. \\
0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 1. & 0. & 0. & 0. & 0. \\
0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 1. & 0. & 0. & 0. \\
0. & 0. & 0. & 0. & 0. & 0. & 0. & -a_o & -a_1 & -a_2 & 0. & 0. & 0. \\
\hline
0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & a_{g_1}' & 0. & 0. \\
0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & a_{g_2}' & a_g & 0. \\
0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & q_g
\end{array}\right] \qquad (5\text{-}11)$$

In Equations (5-11) and (5-12), the dimensional stability derivatives "X", "Z", and "M" refer to body forces acting along the X- and Z-axes (forward and down axes) and the pitching moment about the Y-axis, (axis emanating from aircraft center of mass toward the right wing) respectively. The subscript identifies the state with respect to which the derivative is taken. The term "g" is the acceleration due to gravity. The subscripts HT and TEF refer to deflections of the horizontal tail and trailing-edge-flap, respectively. The terms α_o, u_o and w_o refer to trim values of angle of attack, and velocity components along the X- and Z-axes, respectively. A trim condition is a steady-state level flight condition about which the longitudinal dynamic equations are linearized. For this case, the trim condition is chosen to be steady level flight at a Mach number of 0.6 and an altitude of 10,000 feet.

Table (5-1) gives the values needed to define the thirteen-state truth model. Note that the value of α_w given in the gust model coefficients is for "Level 1" or light to moderate air turbulence. This level of air turbulence has a high probability of being encountered in flight.

From Chapter II, the truth model is of the form

$$\underline{\dot{x}}_t(t) = F_t\, \underline{x}_t(t) + B_t\, \underline{u}(t) + G_t\, \underline{w}_t(t) \qquad (5\text{-}13)$$

The control vector, $\underline{u}$, is given by

$$\underline{u} = \begin{bmatrix} \delta_{HT_{com}}(t) \\ \\ \delta_{TEF_{com}}(t) \end{bmatrix} \qquad (5\text{-}14)$$

Table 5-1

AFTI/F-16 Data for M = 0.6, H = 10,000 Feet

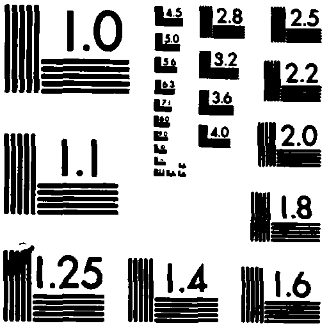
TRIM CONDITIONS	ACTUATOR PARAMETERS	ADDITIONAL PARAMETERS
V_T = 646.4 Ft/Sec	a_o = 102980.0	g = 32.174 Ft/Sec^2
α_o = 2.174 Deg	T_S = 20.2 /Sec	b = 30 Ft
u_o = 645.9 Ft/Sec	ξ = 0.736	σ_w = 5.0 Ft/Sec
w_o = 24.52 Ft/Sec	ω_n = 71.4 /Sec	L_w = 1750.0 Ft

LONGITUDINAL STABILITY DERIVATIVES

Z - BODY FORCES	X - BODY FORCES	PITCHING MOMENTS
Z_α = -1.355 /Sec	X_α = 7.56 Ft/Sec^2	M_α = 4.95X10^{-2} /Sec^2
$Z_{\dot{\alpha}}$ = 2.241X10^{-3}	$X_{\dot{\alpha}}$ = -5.5X10^{-2} Ft/Sec	$M_{\dot{\alpha}}$ = -0.1556 /Sec
Z_q = -5.435X10^{-3}	X_q = 0.1334 Ft/Sec	M_q = -0.5376 /Sec
Z_u = -1.215X10^{-4} /Ft	X_u = -6.47X10^{-3} /Sec	M_u = -6.55X10^{-4} /Sec/Ft
$Z\delta_{HT}$ = -0.1335 /Sec/Rad	$X\delta_{HT}$ = 0.8404 Ft/Sec^2/Rad	$M\delta_{HT}$ = -0.1332 /Sec^2/Rad
$Z\delta_{TEF}$ = -0.2365 /Sec/Rad	$X\delta_{TEF}$ = -2.38 Ft/Sec^2/Rad	$M\delta_{TEF}$ = -1.32 /Sec^2/Rad

AD-A138 425   ROBUST FLIGHT CONTROLLERS(U) AIR FORCE INST OF TECH   2/3
WRIGHT-PATTERSON AFB OH SCHOOL OF ENGINEERRNG
J M HOWEY DEC 83 AFIT/GAE/EE/83D-2

UNCLASSIFIED   F/G 12/1   NL



1.0
4.5
5.0
5.6
6.3
7.1
8.0
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

The B_t and G_t matrices are given by Equations (5-15) and (5-16) on the following page. The last element of the G_t matrix is defined to be

$$q_n = a_{g_d} \times a_n \tag{5-17}$$

where a_n and a_{g_d} were defined previously in Equations (5-9d) and (5-9e).

Employing the data of Table (5-1) and Equations (5-11) and (5-12), the F_t matrix is given by Equation (5-18). Equations (5-19) through (5-22) give the other system matrices with their numerical values inserted.

Recall the continuous-time measurement equation is of the form

$$\underline{z}_t(t) = H_t\, \underline{x}_t + \underline{v}_t(t) \tag{5-23}$$

The covariance kernel description, R_t, for the measurement noise, $\underline{v}_t(t)$, (Equation (2-28)) was established using the values given in Reference 12. The results are listed in Equation (5-24).

$$R_t = \begin{bmatrix} 9.52 \times 10^{-8} & 0 & 0 \\ 0 & 2.44 \times 10^{-7} & 0 \\ 0 & 0 & 6.44 \times 10^{-7} \end{bmatrix} \tag{5-24}$$

Recall from Equation (2-120) that a first order approximation for a discrete-time R_{dt}, to be used to design a sampled-data controller is given by

$$R_{dt} = R_t/\Delta t \tag{5-25}$$

The truth model described above will hereafter be referred to as (T13,10,0.6). "T" indicates truth model, 13 refers to the number of

$$B_t = \begin{bmatrix} 0. & 0. \\ 0. & 0. \\ 0. & 0. \\ 0. & 0. \\ 0. & 0. \\ 0. & 0. \\ a_o & 0. \\ 0. & 0. \\ 0. & 0. \\ 0. & a_o \\ 0. & 0. \\ 0. & 0. \\ 0. & 0. \end{bmatrix} \quad (5\text{-}15)$$

$$G_t = \begin{bmatrix} 0. \\ 0. \\ 0. \\ 0. \\ 0. \\ 0. \\ 0. \\ 0. \\ 0. \\ 0. \\ 1. \\ a_n \\ q_n \end{bmatrix} \quad (5\text{-}16)$$

$$F_t = \left[\begin{array}{cccc|cccccc|ccc}
0.0 & 0.0 & 1.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
-1.894\times10^{-3} & -1.357 & .9968 & -1.218\times10^{-4} & -.1338 & 0.0 & 0.0 & -0.237 & 0.0 & 0.0 & 0.0 & -1.358 & -5.447\times10^{-3} \\
2.996\times10^{-4} & .2622 & -.6937 & -6.359\times10^{-4} & -13.30 & 0.0 & 0.0 & -1.283 & 0.0 & 0.0 & 0.0 & .2622 & -.5367 \\
-32.18 & 7.635 & -24.45 & -6.463\times10^{-3} & .8478 & 0.0 & 0.0 & -2.367 & 0.0 & 0.0 & 0.0 & 7.635 & .1337 \\
\hline
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 1.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 1.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & -102980 & -7221 & -125.3 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 1.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 1.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -102980 & -7221 & -125.3 & 0.0 & 0.0 & 0.0 \\
\hline
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -.3694 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -1.271\times10^{-3} & -.3694 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -2.151\times10^{-2} & -6.250 & -16.92
\end{array}\right] \qquad (5\text{-}18)$$

$$B_t = \begin{bmatrix} 0. & 0. \\ 0. & 0. \\ 0. & 0. \\ 0. & 0. \\ 0. & 0. \\ 0. & 0. \\ 102980. & 0. \\ 0. & 0. \\ 0. & 0. \\ 0. & 102980 \\ 0. & 0. \\ 0. & 0. \\ 0. & 0. \end{bmatrix} \quad (5\text{-}19) \qquad G_t = \begin{bmatrix} 0. \\ 0. \\ 0. \\ 0. \\ 0. \\ 0. \\ 0. \\ 0. \\ 0. \\ 0. \\ 1. \\ 8.142\text{x}10^{-3} \\ 0.1378 \end{bmatrix} \quad (5\text{-}20)$$

$$Q_t = [1] \quad (5\text{-}21)$$

$$H_t = \begin{bmatrix} 1. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. \\ 0. & 1. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 1. & 0. \\ 0. & 0. & 1. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. \end{bmatrix} \quad (5\text{-}22)$$

states, 10 refers to the altitude in thousands of feet, and 0.6 to the Mach number for which the equations were computed.

5.3 Reduced Order Truth Model

The truth model of the previous section represents an unobservable system because a measurement of the fourth state, u, is unavailable. In practice, it is rarely desirable to feed back this state through a controller. Therefore, the fourth state is deleted from the truth model before any further simplification to yield models upon which to base a controller/ estimator design. In addition, all terms involving $\dot{\alpha}$ stability derivations are ignored because their effects on the terms of Equation (5-19) are negligible. Incorporating these derivatives causes the values for the F_t matrix to change only slightly, except for the terms involving, u, which were deleted. Thus, the modifications listed in Equations (5-12a), (5-12b) and (5-12c) are not necessary. Equation (5-12d) now becomes

$$F_t(12,J) = F_t'(12,J) + a_{g_d} \times F_t'(11,J) \qquad (5\text{-}26)$$

The matrices needed to define the twelve-state truth model are given in Equations (5-27) through (5-33). Note that R_t was defined previously in Equation (5-24).

The twelve-state truth model based on linearization about a trim condition at 10000 feet and Mach 0.6 will be referred to as (T12,10,0.6).

5.4 Controller Design Model

The reduced order controller is formed by modelling the AFTI/F-16 actuator dynamics as first-order lags instead of third-order systems as

$$\underline{x}_t = \begin{bmatrix} \theta(t) \\ \alpha(t) \\ q(t) \\ \delta_{HT}(t) \\ \delta_{HT_1}(t) \\ \delta_{HT_2}(t) \\ \delta_{TEF}(t) \\ \delta_{TEF_1}(t) \\ \delta_{TEF_2}(t) \\ \alpha_g'(t) \\ \alpha_{g}^{\cdot}(t) \\ q_g(t) \end{bmatrix} \tag{5-27}$$

$$\underline{u}(t) = \begin{bmatrix} \delta_{HT}(t)_{com} \\ \delta_{TEF_{com}}(t) \end{bmatrix} \tag{5-28}$$

$$F_t = \left[\begin{array}{ccc|cccccc|ccc}
0. & 0. & 1. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. \\
-1.889\text{X}10^{-3} & -1.355 & 0.9946 & -0.1335 & 0. & 0. & -0.2365 & 0. & 0. & 0. & -1.355 & -5.435\text{X}10^{-3} \\
0. & 4.95\text{X}10^{-2} & -0.5376 & -13.32 & 0. & 0. & -1.32 & 0. & 0. & 0. & 4.95\text{X}10^{-2} & -0.5376 \\
\hline
0. & 0. & 0. & 0. & 1. & 0. & \multicolumn{1}{|c}{0.} & 0. & 0. & 0. & 0. & 0. \\
0. & 0. & 0. & 0. & 0. & 1. & \multicolumn{1}{|c}{0.} & 0. & 0. & 0. & 0. & 0. \\
0. & 0. & 0. & -102980 & -7221. & -125.3 & \multicolumn{1}{|c}{0.} & 0. & 0. & 0. & 0. & 0. \\
\cline{4-9}
0. & 0. & 0. & 0. & 0. & 0. & \multicolumn{1}{|c}{0.} & 1. & 0. & 0. & 0. & 0. \\
0. & 0. & 0. & 0. & 0. & 0. & \multicolumn{1}{|c}{0.} & 0. & 1. & 0. & 0. & 0. \\
0. & 0. & 0. & 0. & 0. & 0. & \multicolumn{1}{|c}{-102980.} & -7221. & -125.3 & 0. & 0. & 0. \\
\hline
0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & -0.3694 & 0. & 0. \\
0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & -1.271\text{X}10^{-3} & -0.3694 & 0. \\
0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & -2.151\text{X}10^{-2} & -6.25 & -16.92
\end{array}\right] \qquad (5-29)$$

$$B_t = \begin{bmatrix} 0. & 0. \\ 0. & 0. \\ 0. & 0. \\ 0. & 0. \\ 0. & 0. \\ 102980. & 0. \\ 0. & 0. \\ 0. & 0. \\ 0. & 102980. \\ 0. & 0. \\ 0. & 0. \\ 0. & 0. \end{bmatrix} \quad (5\text{-}30) \qquad G_t = \begin{bmatrix} 0. \\ 0. \\ 0. \\ 0. \\ 0. \\ 0. \\ 0. \\ 0. \\ 0. \\ 1. \\ 8.142\text{x}10^{-3} \\ 0.1378 \end{bmatrix} \quad (5\text{-}31)$$

$$Q_t = [1] \qquad (5\text{-}32)$$

$$H_t = \begin{bmatrix} 1. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. \\ 0. & 1. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 1. & 0. \\ 0. & 0. & 1. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. \end{bmatrix} \quad (5\text{-}33)$$

given in Equation (5-1). The transfer function for a first-order lag with a coefficient of $1/T_s$, i.e., a lag time of T_s, is given by

$$\frac{\delta(s)}{\delta_{com}(s)} = \frac{1/T_s}{s + 1/T_s} \tag{5-34}$$

In state space representation, the above equation becomes

$$\dot{\delta}(t) = (-1/T_s)\ \delta(t) + (1/T_s)\ \delta_{com}(t) \tag{5-35}$$

where T_s is the time constant for the actuator. Again, δ_{com} is the command surface deflection and δ is the surface deflection output from the actuator. A good fit to the frequency response of the actuators yields $1/T_s = 20.2$ (Ref 27). Equation (5-35) then becomes

$$\dot{\delta}(t) = -20.2\ \delta(t) + 20.2\ \delta_{com}(t) \tag{5-36}$$

Making this modification to the twelve-state truth model of Section 5.3 yields an eight-state controller design model. The matrices needed to define the controller model are given in Equations (5-37) through (5-43).

This eight-state controller model will be referred to as (C8,10,0.6).

5.5 Truth Models at Off-Design Conditions

To evaluate the effectiveness of the techniques previous described for robustifying the Kalman filter, the performance of the controllers are to be evaluated at flight conditions other than that for which the controllers were designed, as well as by using truth models of higher order than the design model. This is accomplished by using a truth model defined at an off-design condition in the performance analysis described in Chapter III. The following sections contain one such truth model.

$$\underline{x}(t) \quad \begin{bmatrix} \theta(t) \\ \alpha(t) \\ q(t) \\ \delta_{HT}(t) \\ \delta_{TEF}(t) \\ \alpha_g'(t) \\ \alpha_g(t) \\ q_g(t) \end{bmatrix} \quad (5\text{-}37) \qquad \underline{u}(t) = \begin{bmatrix} \delta_{HT_{com}}(t) \\ \delta_{TEF_{com}}(t) \end{bmatrix} \qquad (5\text{-}38)$$

$$F = \left[\begin{array}{ccc|cc|ccc}
0. & 0. & 1. & 0. & 0. & 0. & 0. & 0. \\
-1.889\text{X}10^{-3} & -1.355 & 0.9946 & -0.1335 & -0.2365 & 0. & -1.355 & -5.435\text{X}10^{-3} \\
0. & 4.95\text{X}10^{-2} & -0.5376 & -13.32 & -1.32 & 0. & 4.95\text{X}10^{-2} & -0.5376 \\
\hline
0. & 0. & 0. & -20.2 & 0. & 0. & 0. & 0. \\
0. & 0. & 0. & 0. & -20.2 & 0. & 0. & 0. \\
\hline
0. & 0. & 0. & 0. & 0. & -0.3694 & 0. & 0. \\
0. & 0. & 0. & 0. & 0. & -1.271\text{X}10^{-3} & -0.3694 & 0. \\
0. & 0. & 0. & 0. & 0. & -2.151\text{X}10^{-2} & -6.25 & -16.92
\end{array}\right] \qquad (5\text{-}39)$$

$$B = \begin{bmatrix} 0. & 0. \\ 0. & 0. \\ 0. & 0. \\ 20.2 & 0. \\ 0. & 20.2 \\ 0. & 0. \\ 0. & 0. \\ 0. & 0. \end{bmatrix} \quad (5\text{-}40) \qquad G = \begin{bmatrix} 0. \\ 0. \\ 0. \\ 0. \\ 0. \\ 1. \\ 8.142\text{x}10^{-3} \\ 0.1378 \end{bmatrix} \quad (5\text{-}41)$$

$$Q = [1] \qquad (5\text{-}42)$$

$$H = \begin{bmatrix} 1. & 0. & 0. & 0. & 0. & 0. & 0. & 0. \\ 0. & 1. & 0. & 0. & 0. & 0. & 1. & 0. \\ 0. & 0. & 1. & 0. & 0. & 0. & 0. & 0. \end{bmatrix} \qquad (5\text{-}43)$$

5.5.1 Truth Model (T12,20,0.6)

This truth model is defined for an altitude of 20,000 feet and a Mach number of 0.6. The values of parameters and stability derivatives needed to form the system matrices are given in Table (5-2). The truth model G_t and F_t matrices are given in Equation (5-44) and (5-45).

$$G_t = \begin{bmatrix} 0. \\ 0. \\ 0. \\ 0. \\ 0. \\ 0. \\ 0. \\ 0. \\ 0. \\ 1. \\ 7.052\text{X}10^{-3} \\ .1591 \end{bmatrix} \qquad (5\text{-}44)$$

5.6 Summary

This chapter has presented the model which will be used to design LQG regulators as in Chapter II and PI controllers as in Chapter III for the AFTI/F-16. The robustification methods detailed in Chapters II and IV will be applied to the model, and the effect of this will be examined by performing covariance analysis as described in Chapter II. The results of this study are presented in the following chapter.

Table 5-2

AFTI/F-16 Data for M = 0.6, H = 20,000 Feet

TRIM CONDITIONS	ACTUATOR PARAMETERS	ADDITIONAL PARAMETERS
V_T = 622.15 Ft/Sec	a_o = 102980.	g = 32.174 Ft/Sec^2
α_o = 3.152 Deg	T_S = 20.2 /Sec	b = 30 Ft
u_o = 621.2 Ft/Sec	ξ = 0.736	σ_w = 2.5 Ft/Sec
w_o = 34.2 Ft/Sec	ω_n = 71.4 /Sec	L_w = 1750. Ft

LONGITUDINAL STABILITY DERIVATIVES

Z - BODY FORCES	X - BODY FORCES	PITCHING MOMENTS
Z_α = -1.006 /Sec	X_α = 23.79 Ft/Sec^2	M_α = -1.218 /Sec^2
Z_α = $1.66X10^{-3}$	X_α = $-5.69X10^{-2}$ Ft/Sec	M_α = -0.1068 /Sec
Z_q = $-3.97X10^{-3}$	X_q = 0.136 Ft/Sec	M_q = -0.384 /Sec
Z_u = $-1.48X10^{-4}$ /Ft	X_u = $-4.827X10^{-3}$ /Sec	M_u = $-7.09X10^{-4}$ /Sec/Ft
$Z\delta_{HT}$ = $-9.18X10^{-2}$ /Sec/Rad	$X\delta_{HT}$ = 0.711 Ft/Sec^3/Rad	$M\delta_{HT}$ = -8.95 /Sec^2/RAD
$Z\delta_{TEF}$ = -0.1689 /Sec/Rad	$X\delta_{TEF}$ = -2.165 Ft/Sec^2/RAD	$M\delta_{TEF}$ = -0.7736 /Sec^2/Rad

$$F_t = \left[\begin{array}{ccc|ccc|ccc|ccc}
0. & 0. & 1. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. \\
-1.510\times10^{-4} & -1.377 & 0.9952 & -0.121 & 0. & 0. & -0.220 & 0. & 0. & 0. & -1.397 & -4.83\times10^{-3} \\
0. & -0.8540 & -0.470 & -16.10 & 0. & 0. & -4.18 & 0. & 0. & 0. & -0.8540 & -0.470 \\
\hline
0. & 0. & 0. & 0. & 1. & \multicolumn{1}{c|}{0.} & 0. & 0. & 0. & 0. & 0. & 0. \\
0. & 0. & 0. & 0. & 0. & \multicolumn{1}{c|}{1.} & 0. & 0. & 0. & 0. & 0. & 0. \\
0. & 0. & 0. & -102980. & -7221. & \multicolumn{1}{c|}{-125.3} & 0. & 0. & 0. & 0. & 0. & 0. \\
\cline{4-9}
0. & 0. & 0. & 0. & 0. & \multicolumn{1}{c|}{0.} & 0. & 1. & 0. & 0. & 0. & 0. \\
0. & 0. & 0. & 0. & 0. & \multicolumn{1}{c|}{0.} & 0. & 0. & 1. & 0. & 0. & 0. \\
0. & 0. & 0. & 0. & 0. & \multicolumn{1}{c|}{0.} & -102980. & -7221. & -125.3 & 0. & 0. & 0. \\
\hline
0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & -0.4740 & 0. & 0. \\
0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & -7.200\times10^{-4} & -0.4740 & 0. \\
0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & -1.564\times10^{-2} & -10.29 & -21.72
\end{array}\right] \qquad (5\text{-}45)$$

VI. Results

6.1 Introduction

The results of the study of methods to improve the robustness properties of controlled systems are presented in this chapter. The two methods examined are the techniques of tuning the Kalman filter by inputting both stationary white and time-correlated Gaussian noise into the system model at the control entry points.

The results of applying these techniques are presented first for LQG regulators. In the first sections, results are given for a continuous-time system. Then, two approaches for a discrete-time system are examined: discretizing the continuous-time controller using first-order approximations, and designing a sampled-data controller.

Finally, in the last sections, the robustification techniques are applied to a sampled-data PI controller.

6.2 Robust LQG Regulators

Two separate issues of robustness are addressed in the following sections. The first is the idea of robustifying a system against differences between the real world and the finite-dimension, low-order model that is chosen for controller and Kalman filter design. As discussed in Chapter V, the model to represent the real world (the truth model) and the lower-order controller model were purposefully established so that the differences between them occur in a specific high frequency range. As proposed by Doyle and Stein (Ref 6), the stability characteristics of a full-state feedback system can be recovered by adding white noise at the control entry points during the process of filter tuning. The validity of this claim is examined by looking first at a purposefully reduced-order controller evaluated against a truth model of the same dimension with first-order actuator dynamics. That is, the low-order controller model is evaluated as if it were a perfect representation of the real-world system. The performance of this controller is the best that can be expected with a Kalman filter to estimate the states. Then, the higher-order actuator dynamics are introduced into the truth model to demonstrate the effects of ignored states on the performance of the system. Next, the robustification techniques are introduced to attempt to recover the stability characteristics of a reduced order but full-state feedback system. It would be desirable also to examine the performance of a full-state feedback system to evaluate the claim that the stability robustness properties of the filter-based controller will asymptotically approach those of the full-state feedback controller using the Doyle and Stein technique. Unfortunately, the software used to design and evalute LQG regulators does not include the option of designing an LQ full-state controller. An attempt was made to approximate full-state feedback by setting the measurement noise intensities to small values and assuming a measurement was available for each state, but the results were inconclusive. Therefore, performance comparisons are only presented between designs using the robustified or unmodified Kalman filter.

As discussed in Chapter V, the difference between the controller design model and the truth model against which the performance is evaluated lies in the dynamics model for the actuators. Open loop frequency responses for the third-order model and the first-order model are shown in Figure (6-1). It is seen that the frequency responses differ

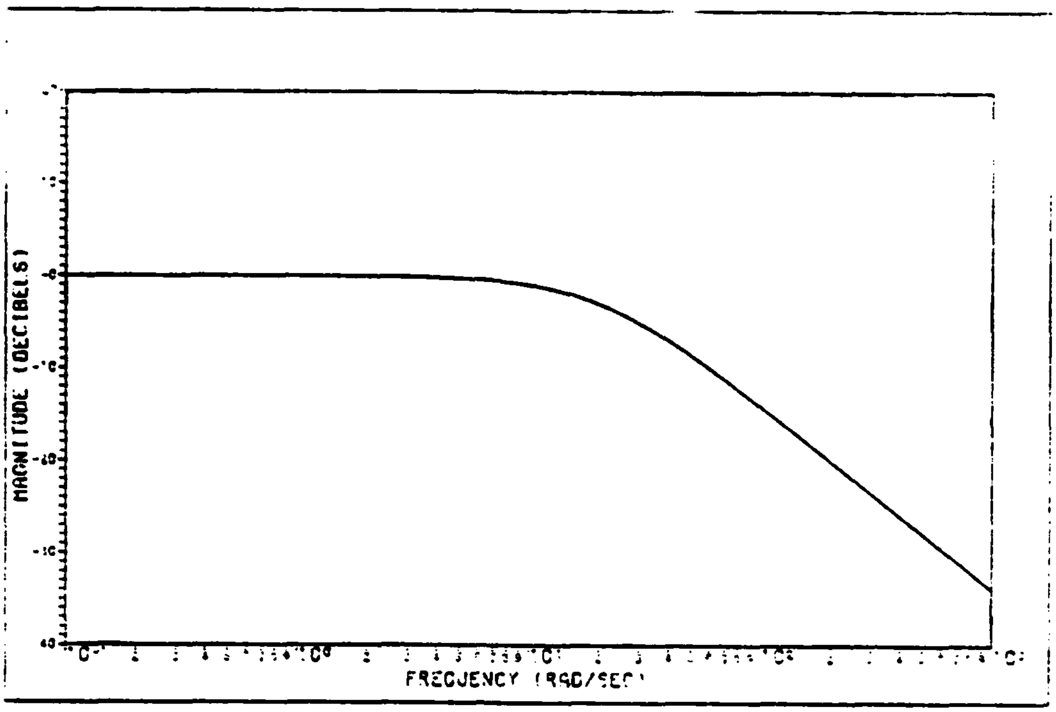


Figure 6-1a: Open Loop Frequency Response for First-Order Actuator Model

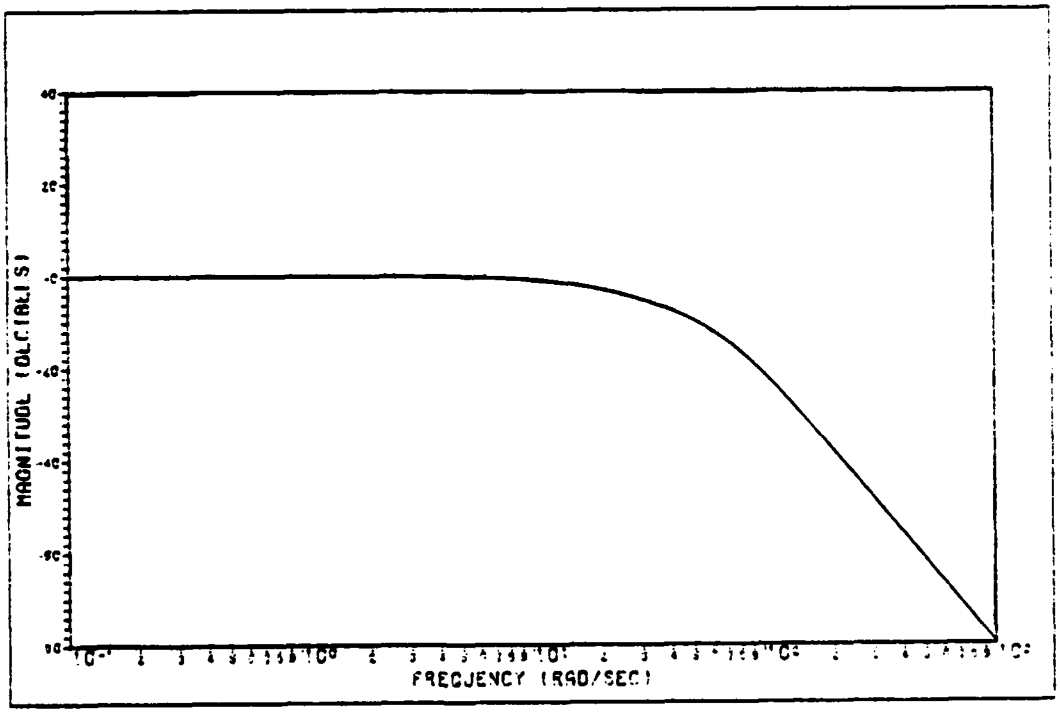


Figure 6-1b: Open-Loop Frequency Response for Third-Order Actuator Model

in the region around 70 rad/sec and beyond, where the complex poles are located in the third-order model. Robustification of the LQG controller based on the reduced-order model can be accomplished by adding white noise to the filter's system model at the point of entry of $\underline{u}$, or, by adding time-correlated noise with the primary power concentrated around the region of the model inadequacy, i.e., around 70 rad/sec and higher.

One first-order and one second-order shaping filter transfer function are considered, each of which concentrates the highest strength of the time-correlated noise in the region where the design model and truth-model actuator dynamics differ. The first-order shaping filter is described by the transfer function

$$\frac{x_u}{w_u} = \frac{s + 0.5}{s + 50} \tag{6-1}$$

A power spectral density plot of the time-correlated noise generated by this shaping filter is shown in Figure (6-2a).

The second-order shaping filter is described by the transfer function

$$\frac{x_u}{w_u} = \frac{s + 0.5}{(s + 50)\ (s + 400)} \tag{6-2}$$

Figure (6-2b) shows the power spectral density function for the time-correlated noise generated by the above shaping filter. The values in Equations (6-1) and (6-2) were chosen by examining power spectral density plots of the time-correlated noise with the poles and zeroes of the shaping filters in different locations. The chosen values generate the

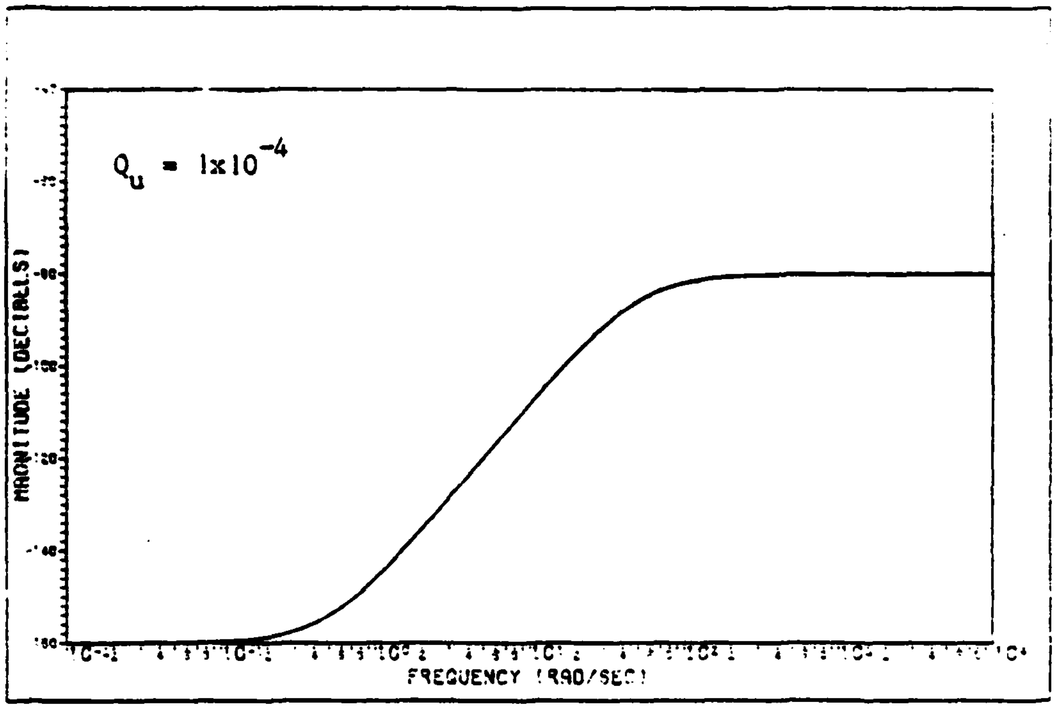


Figure 6-2a: First-Order Colored Noise Process Power Spectral Density

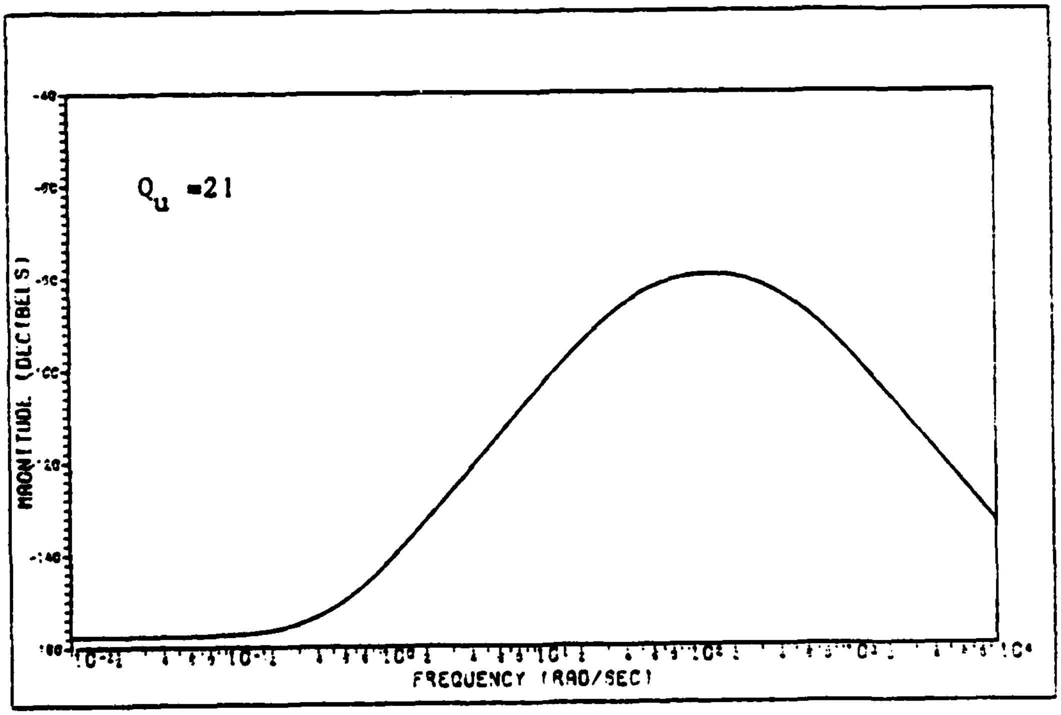


Figure 6-2b: Second-Order Colored Noise Process Power Spectral Density

desired noise, the strength of which is highest in the frequencies where the truth and design model frequency responses differ.

The second robustness issue is concerned with operation of the controlled system at flight conditions other than the design condition. The differences between the real world and the reduced-order design model now occur in other frequency ranges besides those given by the actuator dynamics models. Since these frequency ranges are now unknown, the use of white input noise would be motivated since it injects equal uncertainty into the system model over the entire frequency spectrum. The additional performance degradation or instabilities that arise in going from the design condition to an off-design condition with a Kalman filter in the loop are examined to determine the success of the Kalman filter robustification techniques. That is, if the full-state feedback system (based on a reduced-order design model) is stable, then the LQG system (which may well yield an unstable closed-loop system with the unrobustified filter in the loop) can be stabilized with the addition of white input noise (as claimed by Ref 6) and possibly by colored noise if the robustification is applied over the appropriate frequency ranges.

The cost-weighting matrices used in Equation (2-3) are the same for the continuous-time, discretized, and sampled-data regulators (discretizing (2-3) to yield (2-69) is accomplished as in Ref 24). The values used are based on those given in References 12 and 22, with some iteration on the control weightings to achieve designs with commanded control surface deflections that do not exceed physical limits of the horizontal tail and tailing-edge flap. The state, control and cross-weighting matrices are given by

$$W_{xx} = \begin{bmatrix} 50 & & & -0- \\ & 50 & & \\ & & 150 & \\ -0- & & & -0- \end{bmatrix} \qquad (6\text{-}3)$$

$$W_{xu} = [0] \qquad (6\text{-}4)$$

$$W_{uu} = \begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix} \qquad (6\text{-}5)$$

It is desired to make a direct comparison between the performance of the controlled system with white and time-correlated input noise added to the model upon which the Kalman filter is based, where the maximum intensity of the time-correlated noise is equal to the intensity of the white noise. By equating the maximum magnitudes of the white and time-correlated noise, it can be determined if there are performance benefits in choosing one type of noise over another by comparing the degree of robustification achieved by each. This is accomplished using the frequency domain shaping filter design techniques in Reference 23, which state that

$$PSD_o(s) = G(s)\ G(-s)\ PSD_i(s) \qquad (6\text{-}6)$$

where PSD_i and PSD_o are the power spectral densities of the input and output of a shaping filter, respectively. G(s) is the transfer function of the shaping filter. It is desired to have the maximum magnitude of the right-hand side of Equation (6-6) be equal the magnitude of the white input noise, (the scalar white noise parameter, q^2 of Equation (2-64)). Thus, by setting PSD_o equal to q^2 and solving for the value of PSD_i which makes this true, the value of Q_u in Equation (4-12) can be found for the time-correlated noise. Thus, it is the intensity, Q_u, of the white driving noise, $w_u(t)$ to the shaping filter which will generate a time-correlated noise with a maximum intensity of q^2. Table (6-1) lists the values of Q_u which make the maximum intensity of the time-correlated noise equal to that of the white noise. The values were arrived at using the software described in Reference 20.

Table 6-1

Strength of Dynamic Driving Noise to Shaping Filters

$G(s) = 1$	$G(s) = \frac{s + 0.5}{s + 50}$	$G(s) = \frac{s + 0.5}{(s + 50)(s + 400)}$
q^2	Q_u	Q_u
1×10^{-6}	1×10^{-6}	0.21
1×10^{-4}	1×10^{-4}	21.
1×10^{-2}	1×10^{-2}	2100.

6.2.1 Continuous-Time LQG Regulators at Design Condition

Figure (6-3) shows time histories of the mean and standard deviation of the aircraft state, Θ, for an eight-state controller evaluated against a truth model of the same dimension (the truth model and controller model are identical as given in Section 5.4). A perfectly known initial condition of one degree (or 0.0175 radians) was placed on Θ, and the

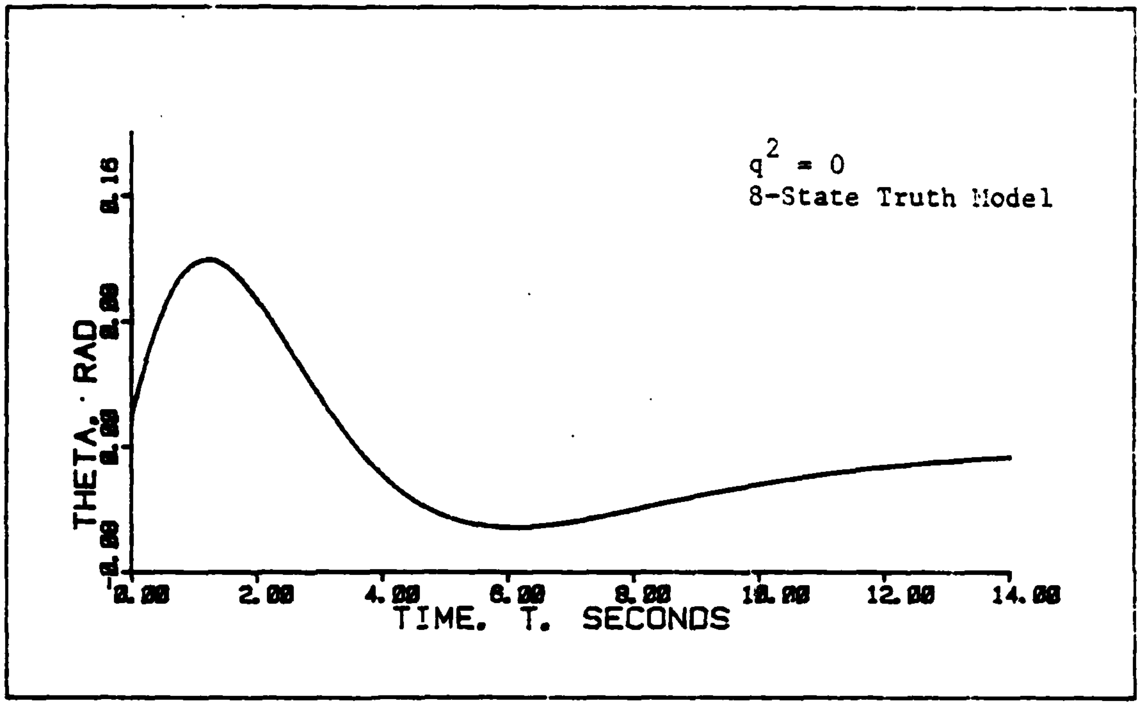


Figure 6-3a: Mean of Θ at the Design Condition

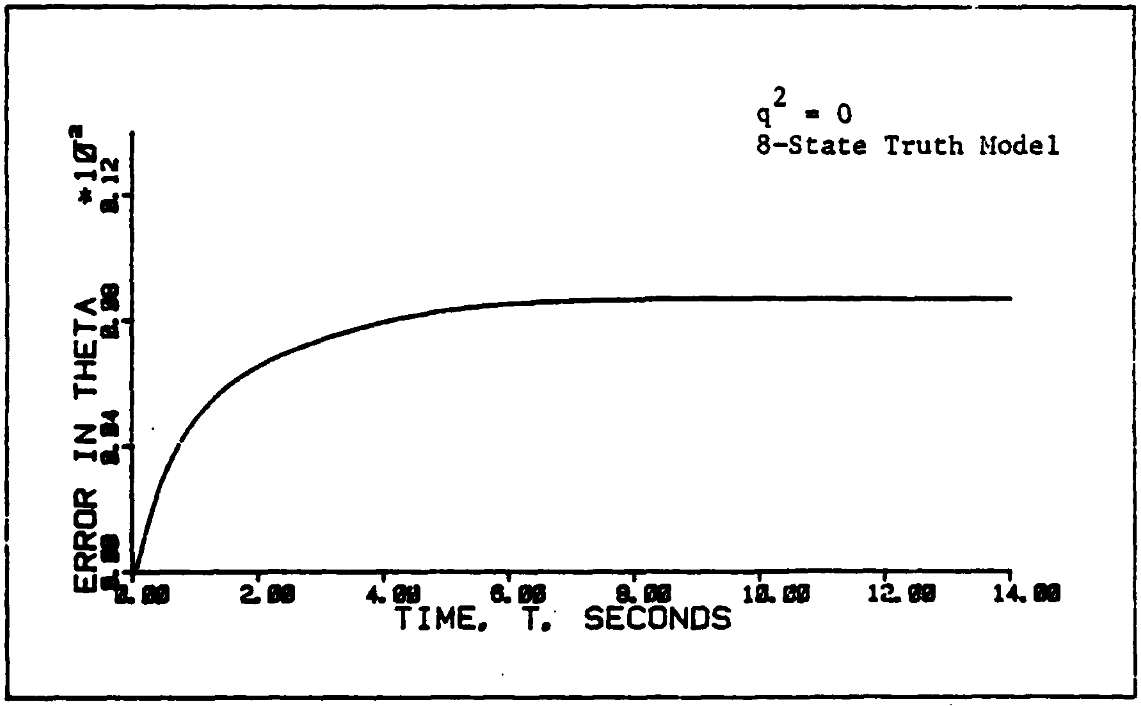


Figure 6-3b: Standard Deviation of Θ at the Design Condition

plot shows the response of Θ to the initial perturbation and to the dynamic driving noise built into the model. The trends are similar for all three aircraft states, therefore only one will be examined in detail in this chapter.

The time response is stable, although slow. The slow response is typical of systems with a Kalman filter to estimate states. The added dynamics of the filter tend to slow down the response. However, the controller does respond to the perturbation and regulate the state back to zero. Figure (6-4) shows the response of the same controller when specific ignored states are now accounted for in the truth model. This figure is the result of a performance analysis when third-order actuator dynamics are included in the twelve-state truth model, while only first-order dynamics are used in the eight-state design model. As can be seen, the system is still stable, bu the mean of Θ response has degraded substantially, especially in steady-state.

Figures (6-5a) and (6-5b) show the response of the same state with a white Gaussian noise of strength $q^2 = 1 \times 10^{-4}$ injected into the system model. The transient time, the overshoot, and the error in the final mean value of Θ have all been substantially improved with the noise addition. The standard deviation of Θ has not changed noticeably from the previous case. The trend for lower and higher $\sim^2$ values is shown in Figures (6-5c,e,d,f). It is seen that noise of a very small intensity (1×10^{-6}) enhances the robustness properties dramatically, and increasing the intensity of the noise only serves to change the transient characteristics of the mean time response and the magnitude of the standard deviation.

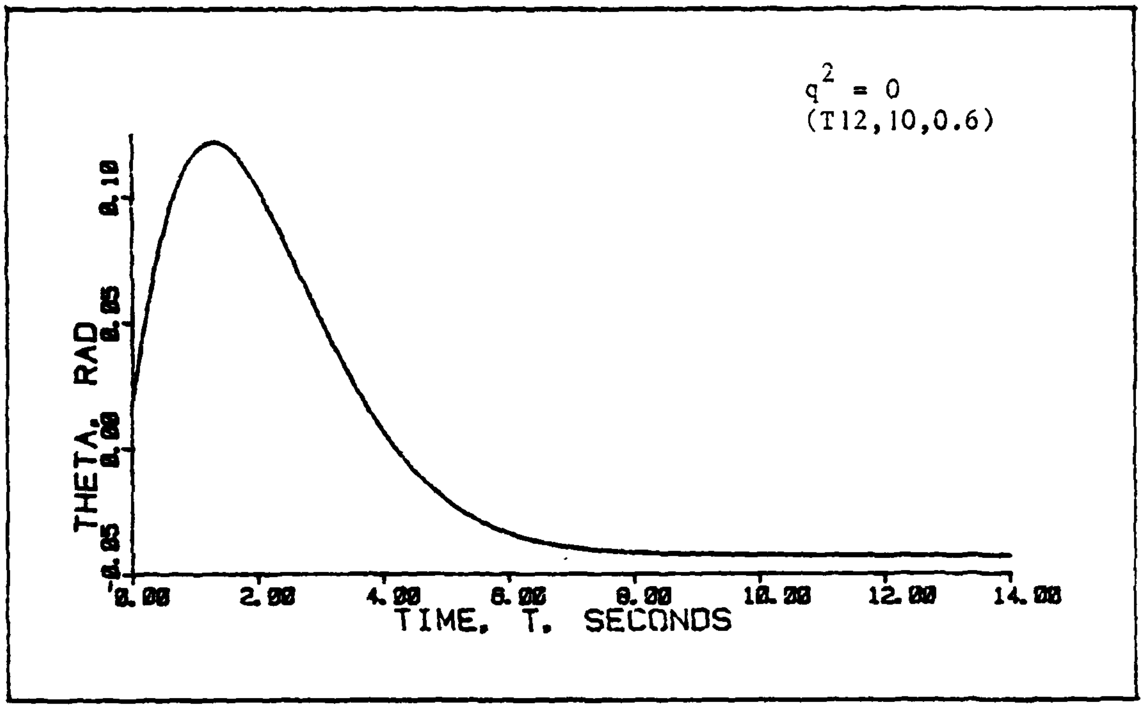


Figure 6-4a: Mean of Θ With no Noise Addition

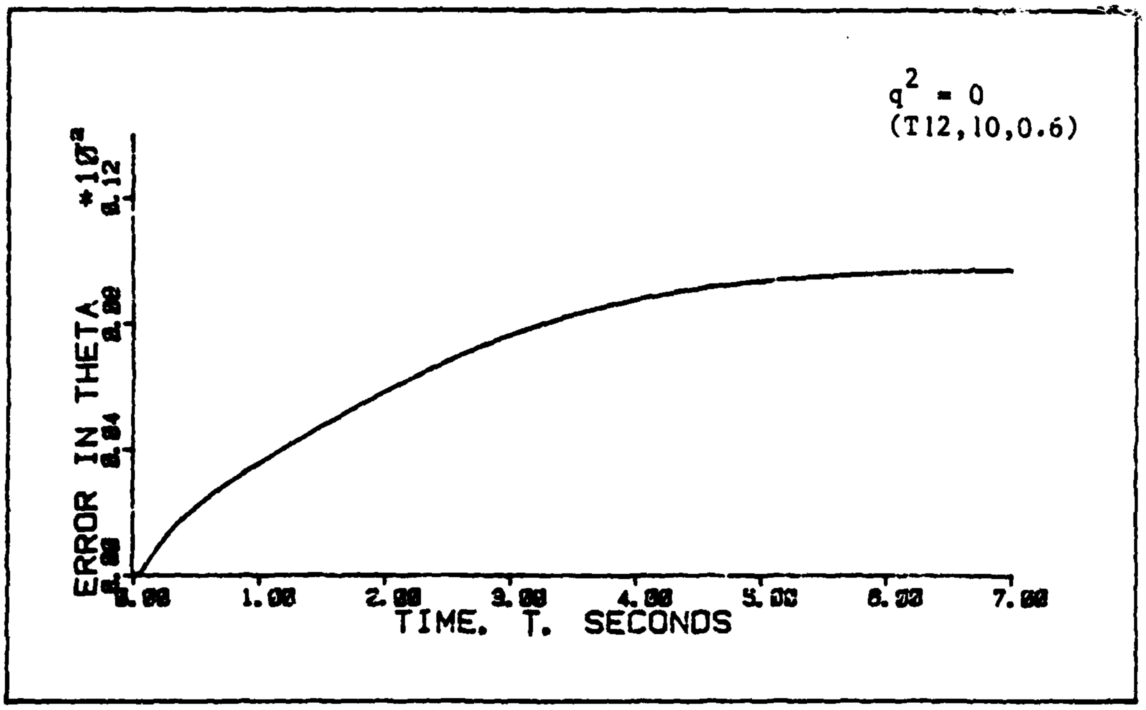


Figure 6-4b: Standard Deviation of Θ With No Noise Addition

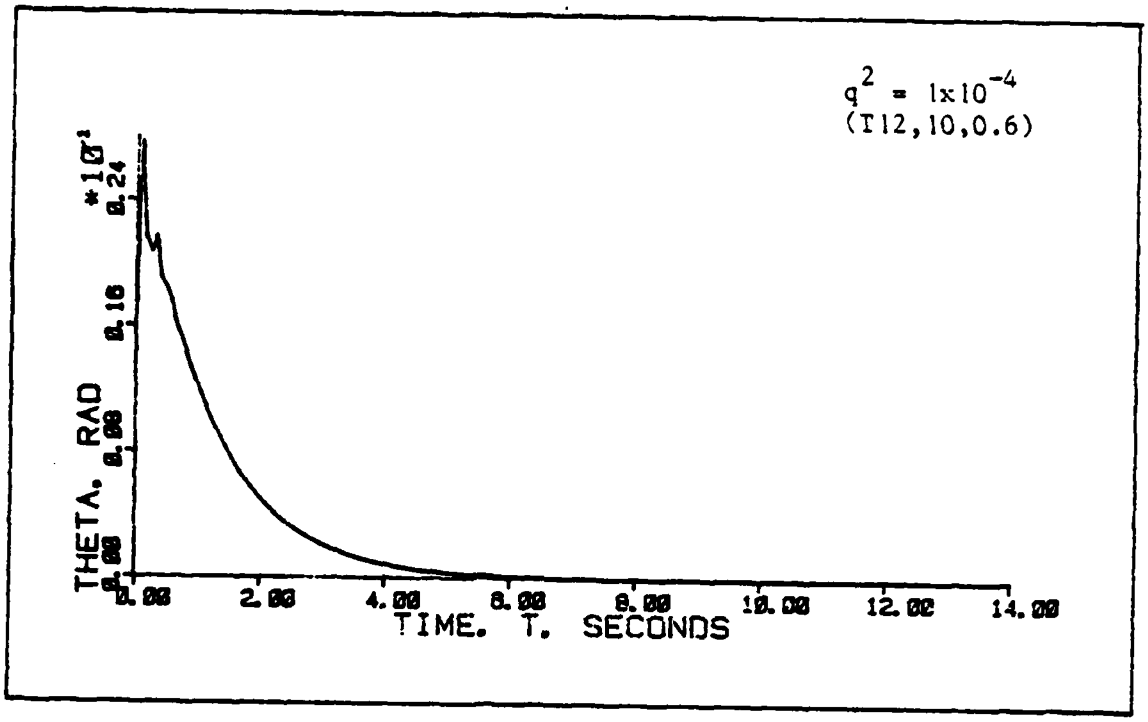


Figure 6-5a: Mean of θ With White Noise Addition

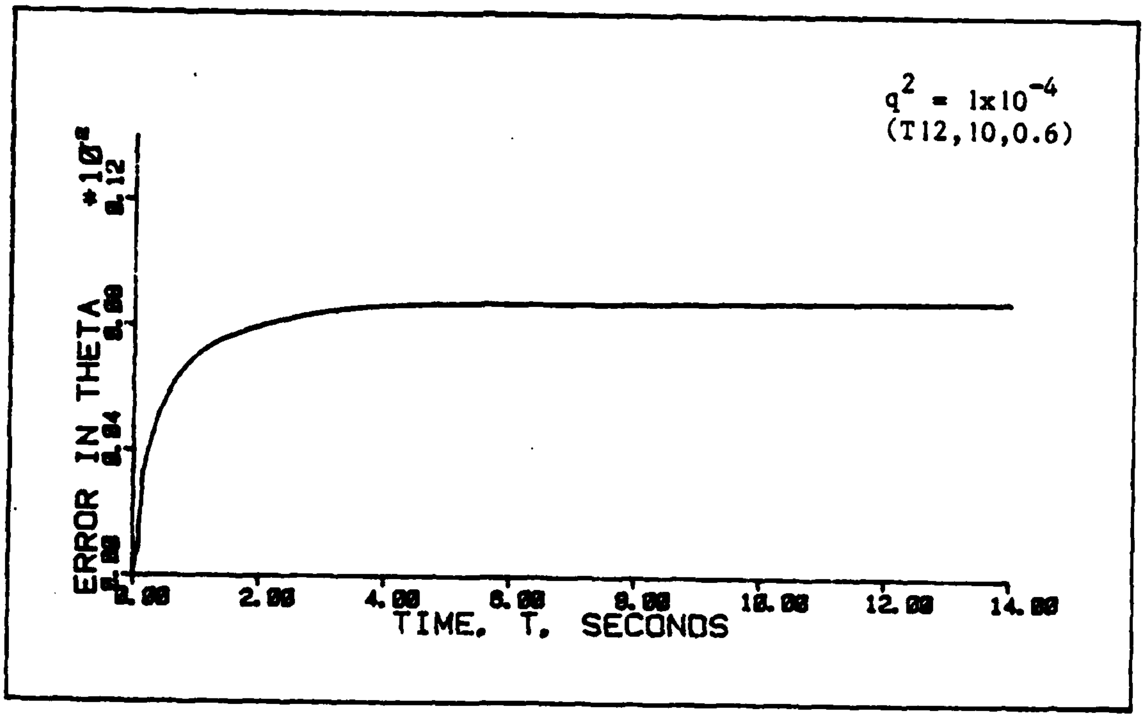


Figure 6-5b: Standard Deviation of θ With White Noise Addition

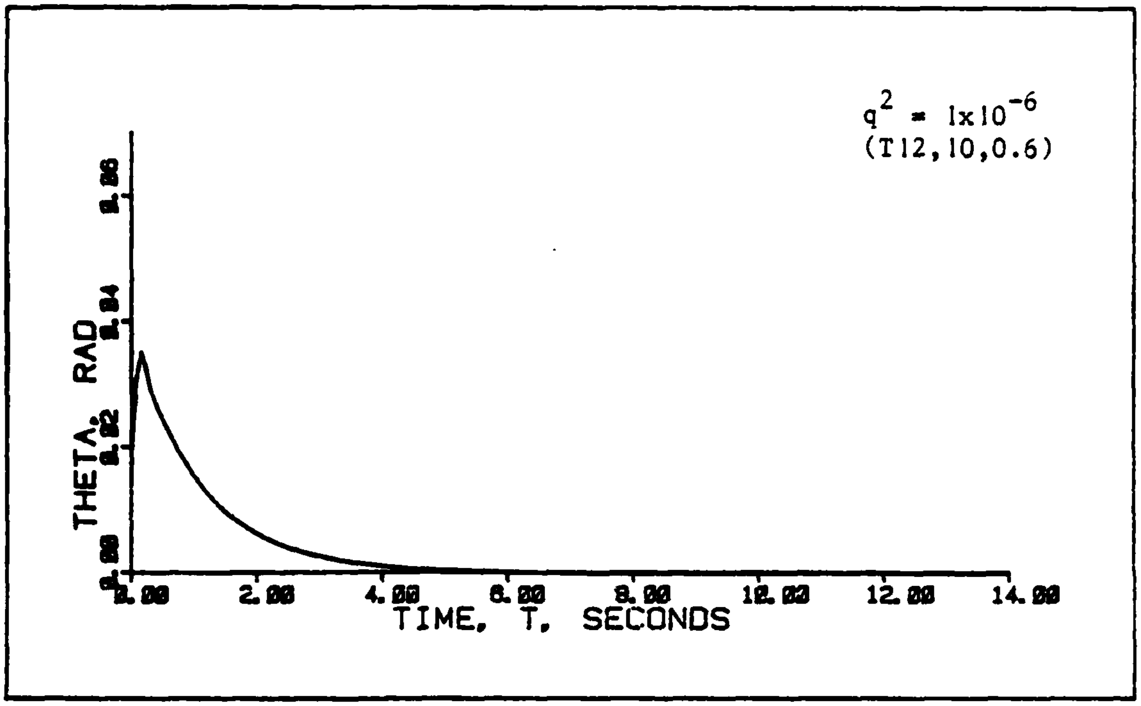


Figure 6-5c: Mean of θ With White Noise Addition

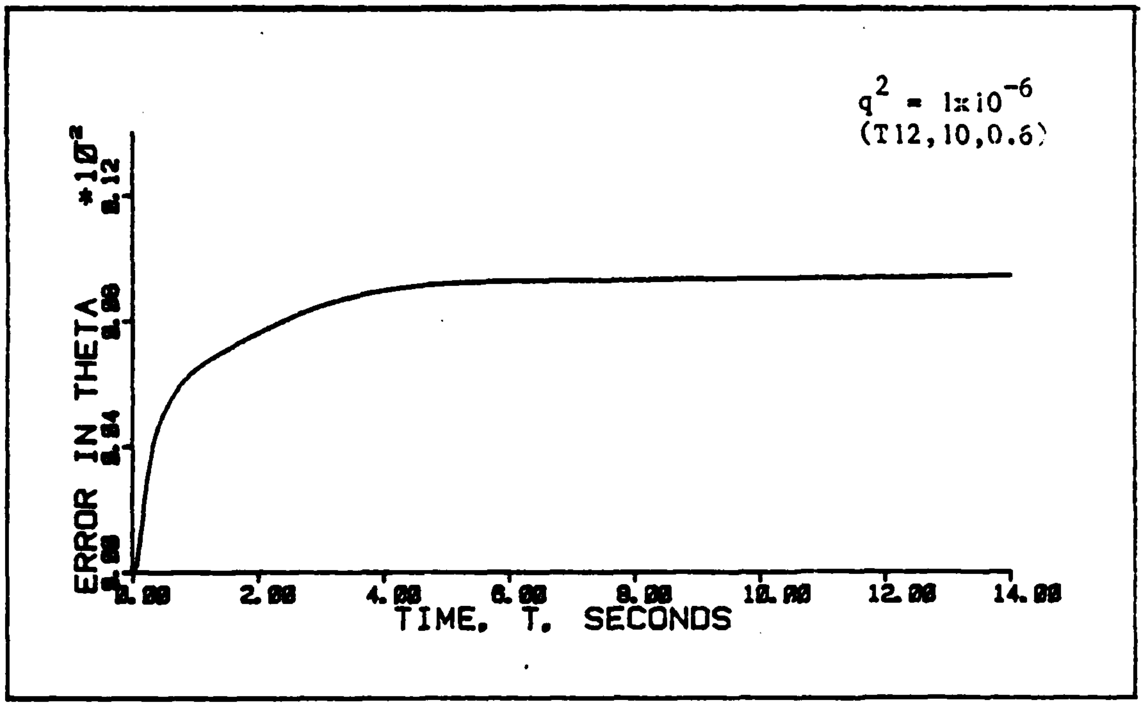


Figure 6-5d: Standard Deviation of θ With White Noise Addition

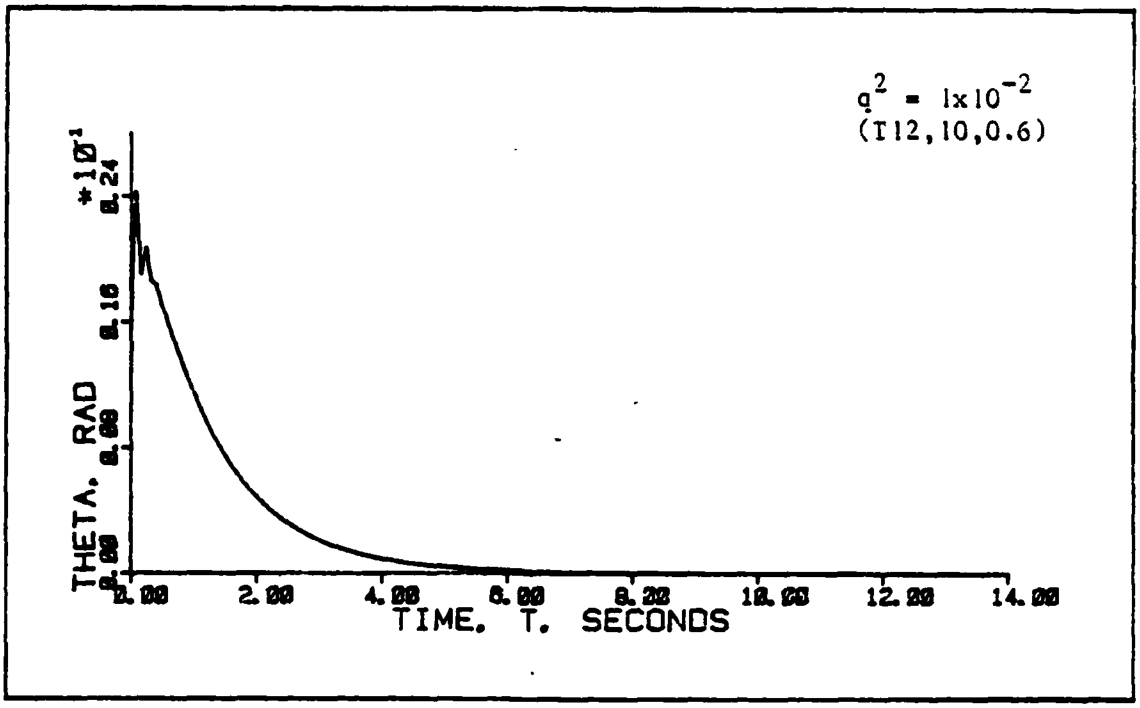


Figure 6-5e: Mean of θ With White Noise Addition

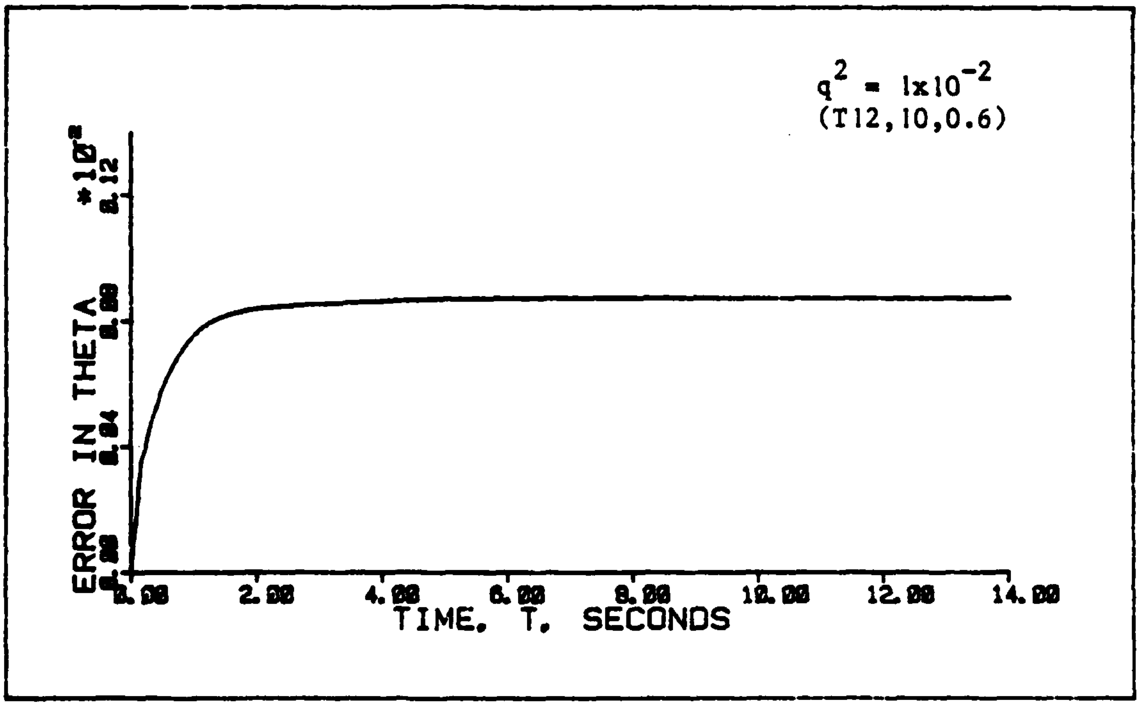


Figure 6-5f: Standard Deviation of θ With White Noise Addition

Then, as shown in Figure (6-6), time-correlated noise generated by a first-order shaping filter is injected into the system model. The maximum power spectral density of the noise is 1 x 10^{-4}. The figure demonstrates that the improvement in the transient time, overshoot, and final value of Θ is as dramatic as for the white noise case. Again, the standard deviation has not changed enough to be discernible in the figure.

Finally, time-correlated noise generated by a second-order shaping filter is injected into the design model. As can be seen in Figure (6-7), the transient time is approximately the same as for the original system, but much slower than the cases above. However, the state is converging to zero, and the overshoot has been reduced to about half of the original value. Any change in the steady-state value of the standard deviation is not noticeable in the figure.

Thus, Figures (6-3) through (6-7) demonstrate that the robustification techniques can recover the stability robustness characteristics that would be expected from a full-state feedback system. The methods of injecting white and first-order colored noise produce very similar improvements, while the second-order colored noise produced less but still noticeable benefits in the time response.

However, it was stated in previous chapters that the addition of noise into the design model will degrade the performance (as measured by the standard deviation of the states or how well known the states are) of the system at the design conditions. Additionally, it was claimed that the performance degradation will be less for colored noise than for white noise. To evaluate this claim, Table (6-2) lists the steady-state values of the standard deviations of all three aircraft states with no input

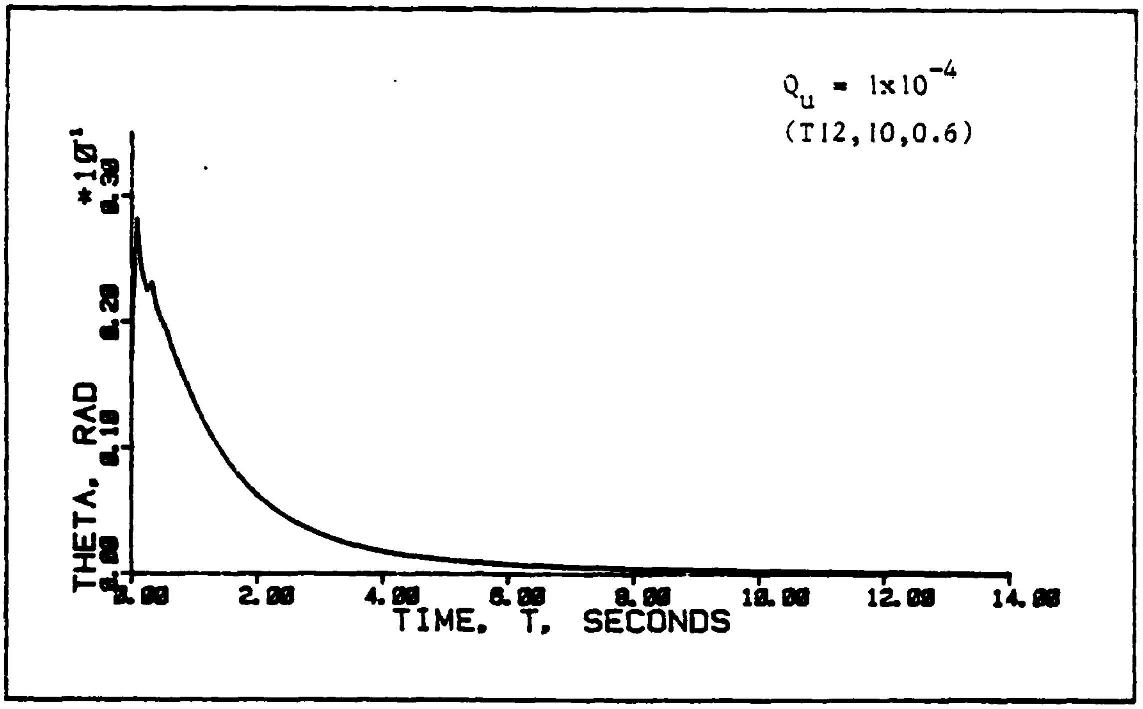


Figure 6-6a: Mean of θ With First-Order Colored Noise Addition

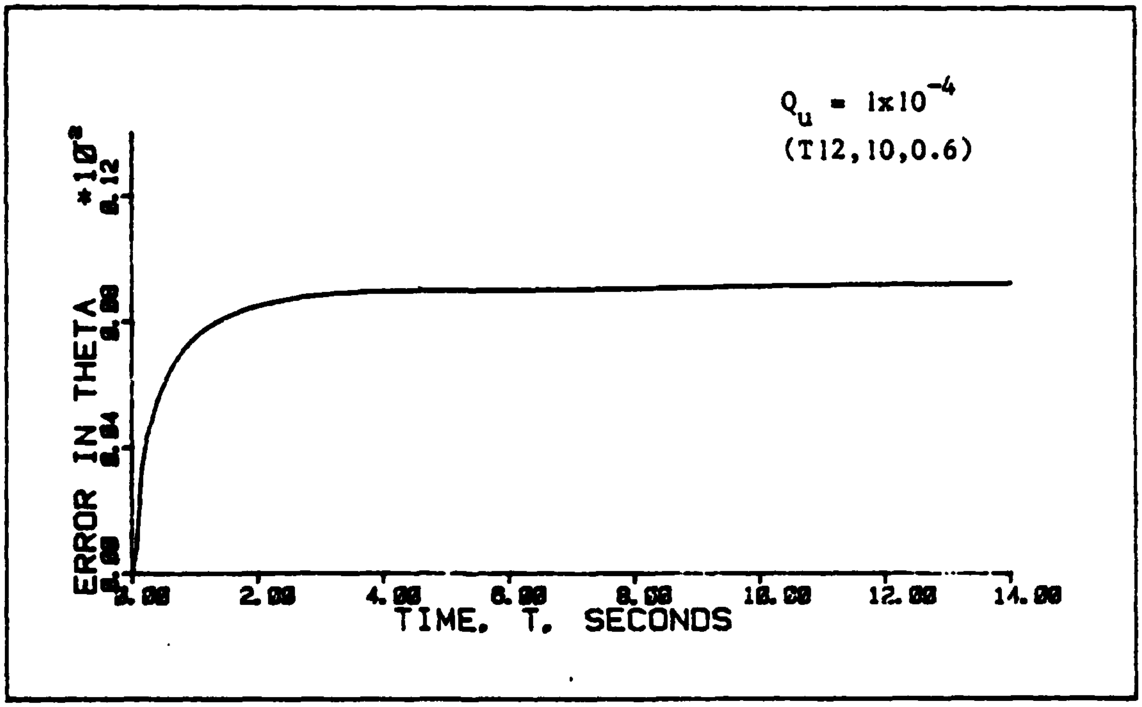


Figure 6-6b: Standard Deviation of θ With First-Order Colored Noise Addition

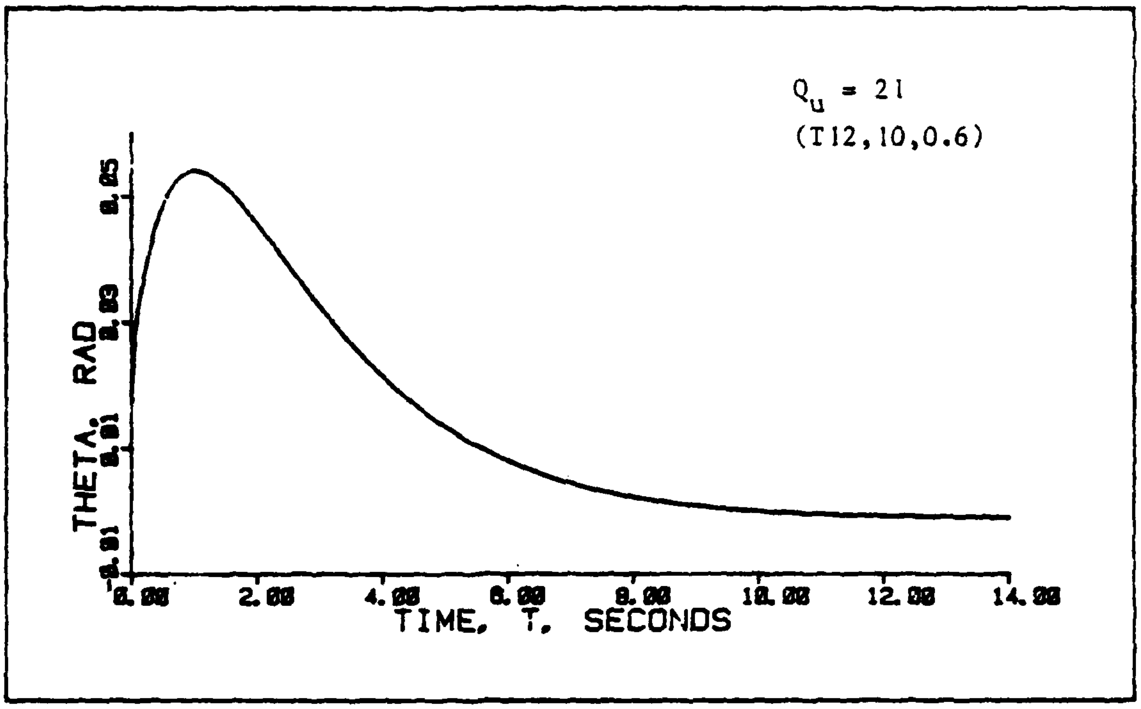


Figure 6-7a: Mean of θ With Second-Order Colored Noise Addition

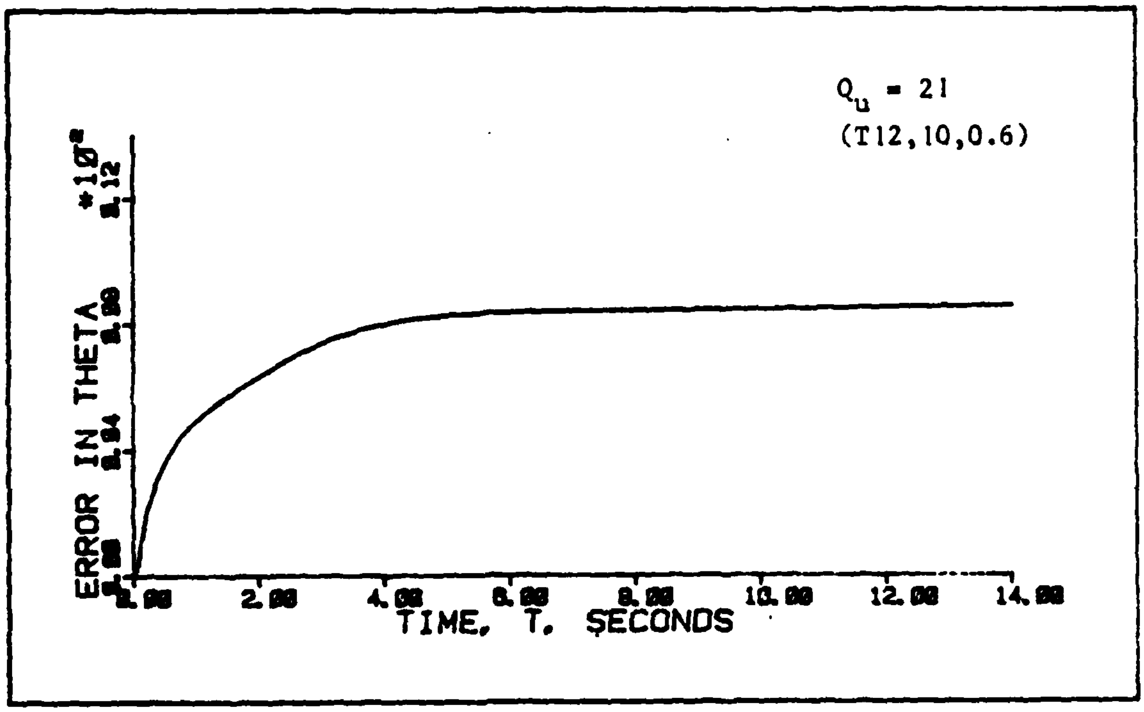


Figure 6-7b: Standard Deviation of θ With Second-Order Colored Noise Addition

noise, white noise, and first-order and second-order time-correlated noise for one value of noise intensity. The values listed are the results of a performance evaluation where the unmodified controller design model and truth model are identical (both are of dimension eight at the design condition). The results shown in the table only partially substantiate the claim that time-correlated noise can minimize the performance degradation. The addition of white noise to the system model increased the standard deviation of all three aircraft states as expected.

It was expected that the addition of first-order colored noise would reduce the standard deviation of the aircraft states to values less than those for white noise, but still larger than the case with no noise addition. Table (6-2) shows that this is not the case. Rather, the standard deviations have increased, if only slightly for Θ and α. Only the pitch rate, q, has decreased as expected. Recall from Figures (6-5a) and (6-6a) that white and first-order colored noise produce similar mean of theta responses. That is, the robustness enhancement is very similar for the two cases. However, even though the added uncertainty is applied over a

Table 6-2

Comparison of Steady-State Deviations of Aircraft States at the Design Condition for a Continuous Time System

	q^2	Q_u	σ_θ	σ_α	σ_q
No noise	0	-	$8.705\text{x}10^{-4}$	$5.035\text{x}10^{-3}$	$1.200\text{x}10^{-3}$
White Noise	$1\text{x}10^{-4}$	-	$9.590\text{x}10^{-4}$	$5.306\text{x}10^{-3}$	$1.978\text{x}10^{-3}$
1st-order Shaping Filter	-	$1\text{x}10^{-4}$	$1.006\text{x}10^{-3}$	$5.823\text{x}10^{-3}$	$1.761\text{x}10^{-3}$
2nd-order Shaping Filter	-	21	$8.704\text{x}10^{-4}$	$5.039\text{x}10^{-3}$	$1.201\text{x}10^{-3}$

more limited frequency range when first-order colored noise is injected into the model, the standard deviations have increased slightly. This is not well understood, except to say that time-correlated noise actually changes the structure of the Kalman filter design model. The added complexity of the design model may overcome any benefits that are realized because the noise was not applied over all frequencies as in the white noise case.

As expected, the performance at design conditions is degraded less by using a second-order filter as opposed to a first-order filter or white noise. However, the greater complexity of the Kalman filter design model (four additional states in this case) is not justified by the performance benefits seen in Table (6-2). In this instance, white noise injected into the system design model provides the desired robustification while not degrading the performance at the design condition substantially.

Table (6-3) presents similar steady-state standard deviation information about the three aircraft states as shown in Table (6-2). However, the values listed here are the results of a performance evaluation of the eight-state controller evaluated against the twelve-state truth model, accounting for higher-order actuator dynamics. An important difference between Tables (6-2) and (6-3) is the effect of adding white input noise on the standard derivation of Θ. At the design condition, adding white noise detunes the filter and results in a performance degradation. However, when third-order actuator dynamics are included in the truth model, adding white noise improves the tuning for the Θ channel, and the standard deviation decreases for this state. Thus, when the performance of the system is evaluated in an environment different from the design condition, the noise addition can result in a performance enhancement. It is seen that for this case, second-order time correlated noise again accomplishes improved filter tuning over first-order and white noise. The improvement, though, is still not substantial enough to justify the added complexity of the Kalman filter design model. White input noise, as stated above, accomplishes the desired robustness enhancement without adding states to the design model.

Table 6-3

Comparison of Steady-State Standard Deviations of Aircraft States with Higher-Order Actuator Dynamics for a Continuous-Time System.

	q_2	Q_u	σ_θ	σ_α	σ_q
No noise	0	-	$8.984\mathrm{x}10^{-4}$	$4.894\mathrm{x}10^{-3}$	$1.189\mathrm{x}10^{-3}$
White Noise	$1\mathrm{x}10^{-4}$	-	$8.850\mathrm{x}10^{-4}$	$5.689\mathrm{x}10^{-3}$	$4.979\mathrm{x}10^{-3}$
1st Order Shaping Filter	-	$1\mathrm{x}10^{-4}$	$9.232\mathrm{x}10^{-4}$	$5.813\mathrm{x}10^{-3}$	$5.092\mathrm{x}10^{-3}$
2nd Order Shaping Filter	-	21	$8.571\mathrm{x}10^{-4}$	$5.168\mathrm{x}10^{-3}$	$1.969\mathrm{x}10^{-3}$

6.2.2 Continuous-Time LQG Regulators at Off-Design Condition

Figure (6-8) shows the results of a performance analysis for the unrobustified system with a Kalman filter at an off-design flight condition (T12,20,0.6) for the eight-state controller design model. As shown in the figure, the system is unstable. The mean of θ is diverging, and the standard deviation is growing with time. In addition, the inputs generated by the controller are growing with time beyond the actual limits of the control surfaces (±23 degrees for the horizontal tail and ±20 degrees for the trailing-edge flap: Ref 27). This is demonstrated in the mean plots of Figure (6-9).

Figures (6-10a) and (6-10b) show the system responses at the same off-design flight condition, except now white noise of strength $q^2 = 1x10^{-4}$ has been injected into the filter's sytem model at the control entry points. The addition of this noise is sufficient to stabilize the system, driving the aircraft state towards zero and the standard deviation of the state to a finite value. Figures (6-10c,d,e,f) demonstrate the trend for a lower and higher value of q. As noted before, stability is recovered with a very low noise strength. Higher values change only transient characteristics and the magnitude of the standard deviation. Figure (6-11) demonstrates that the mean of the commanded controls are no longer exceeding the physical limits of the control surfaces. However, the large initial changes in the command inputs do exceed the actuator rate limits of the control surfaces because there is no weighting on input rates in the cost function for an LQG regulator.

For a time-correlated noise generated by a first-order shaping filter with a maximum intensity of $1x10^{-4}$, the results are similar to those of

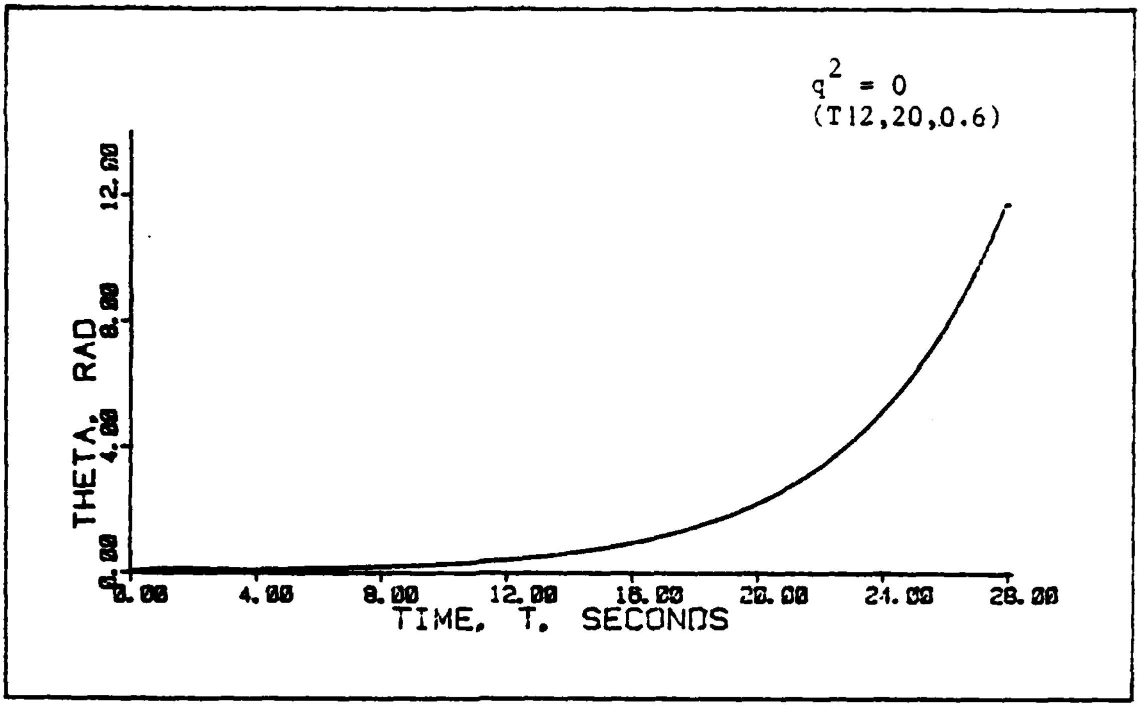


Figure 6-8a: Mean of Theta at Off Design Flight Condition

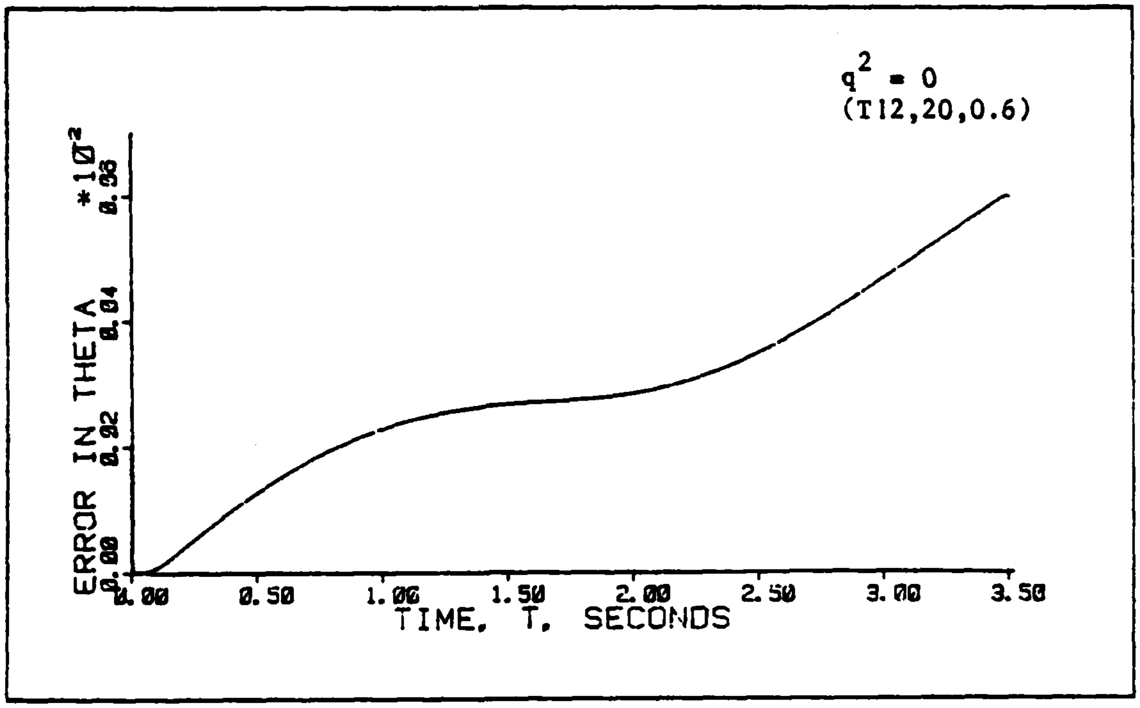


Figure 6-8b: Standard Deviation of θ at Off-Design Flight Condition

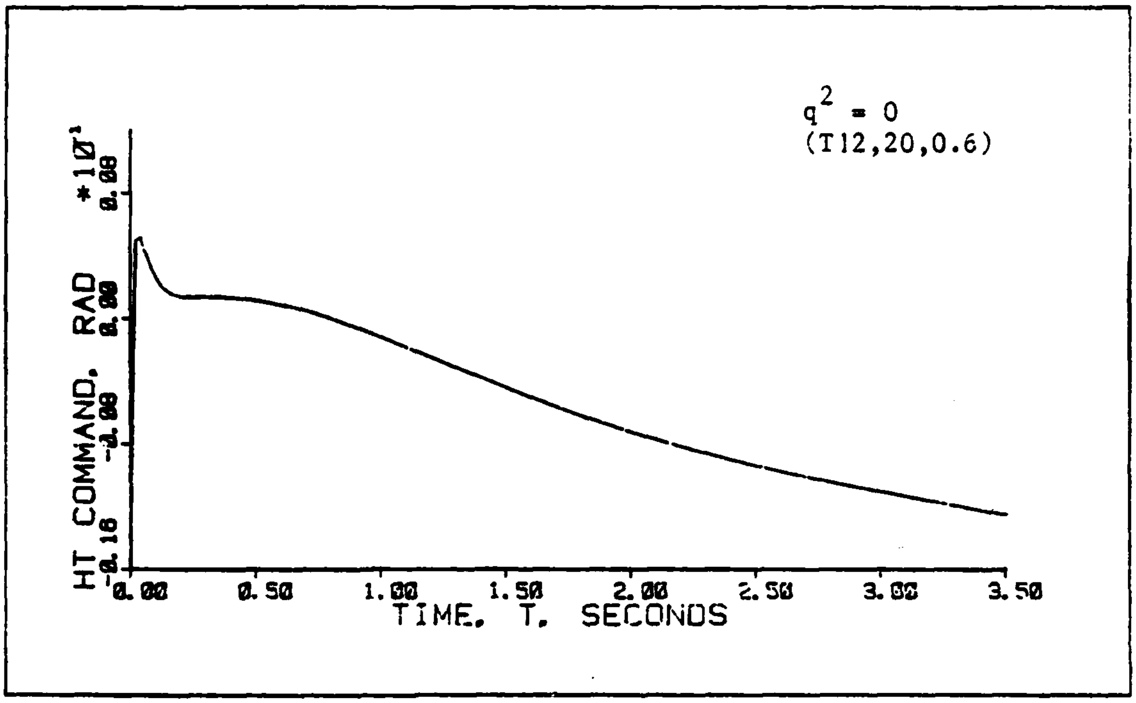


Figure 6-9a: Mean of Horizontal Tail Commanded Deflection

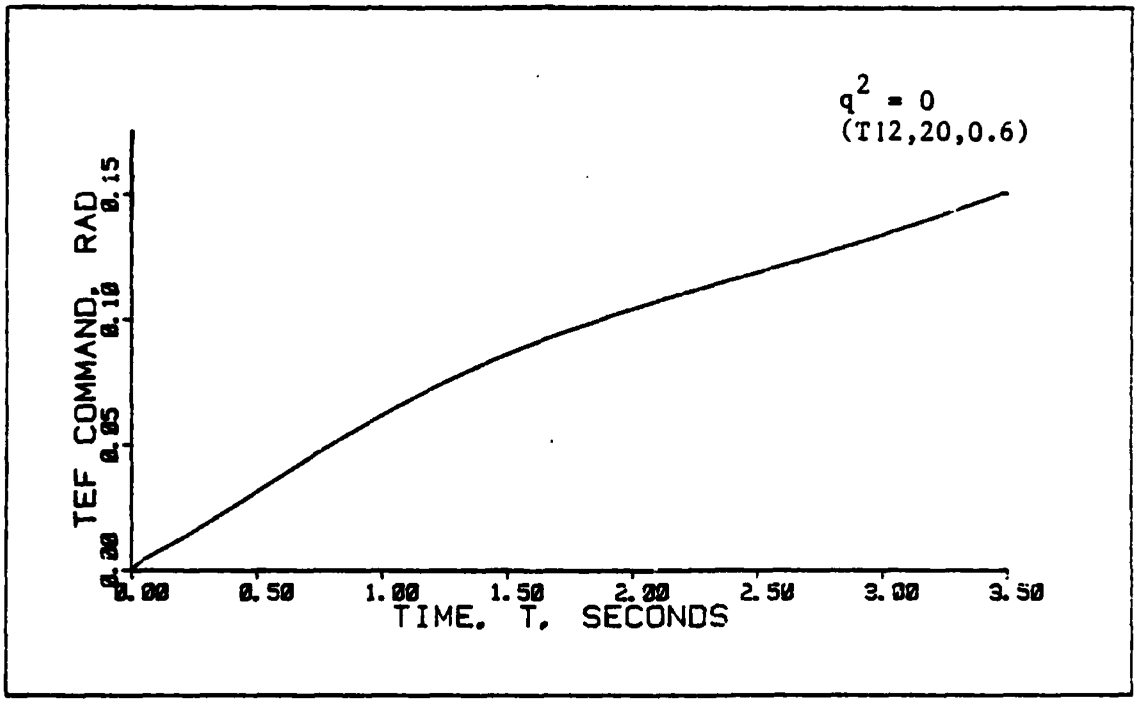


Figure 6-9b: Mean of Trailing Edge Flap Commanded Deflection

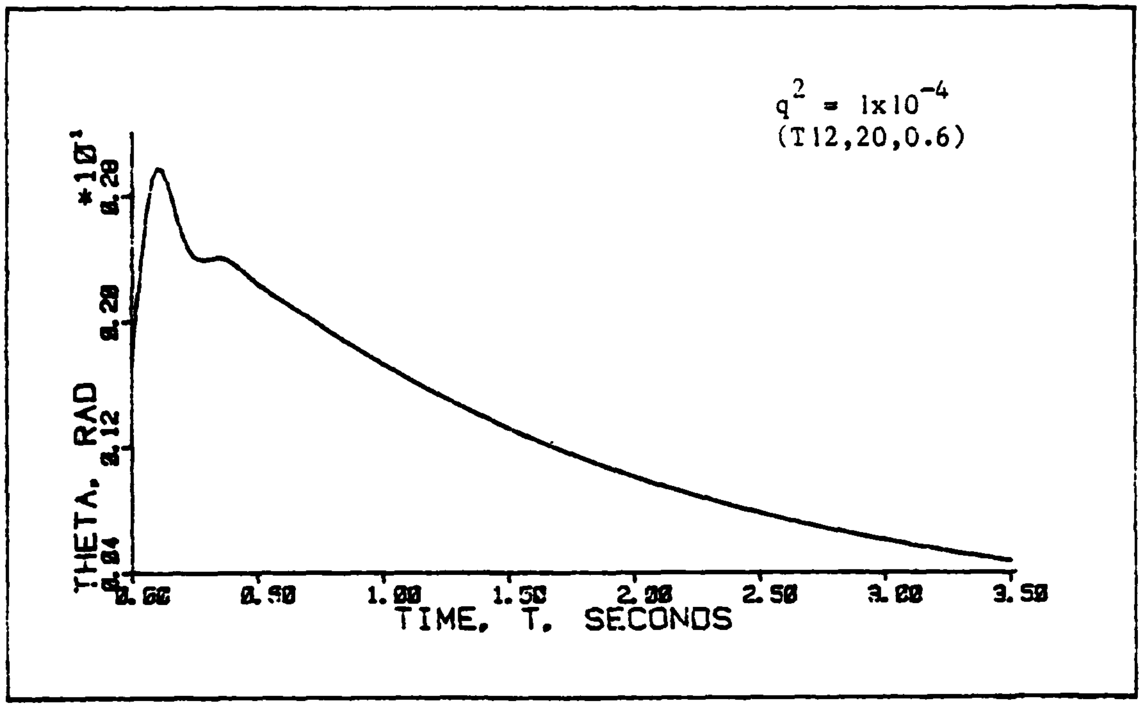


Figure 6-10a: Mean of θ With White Noise Addition at Off-Design Flight Condition

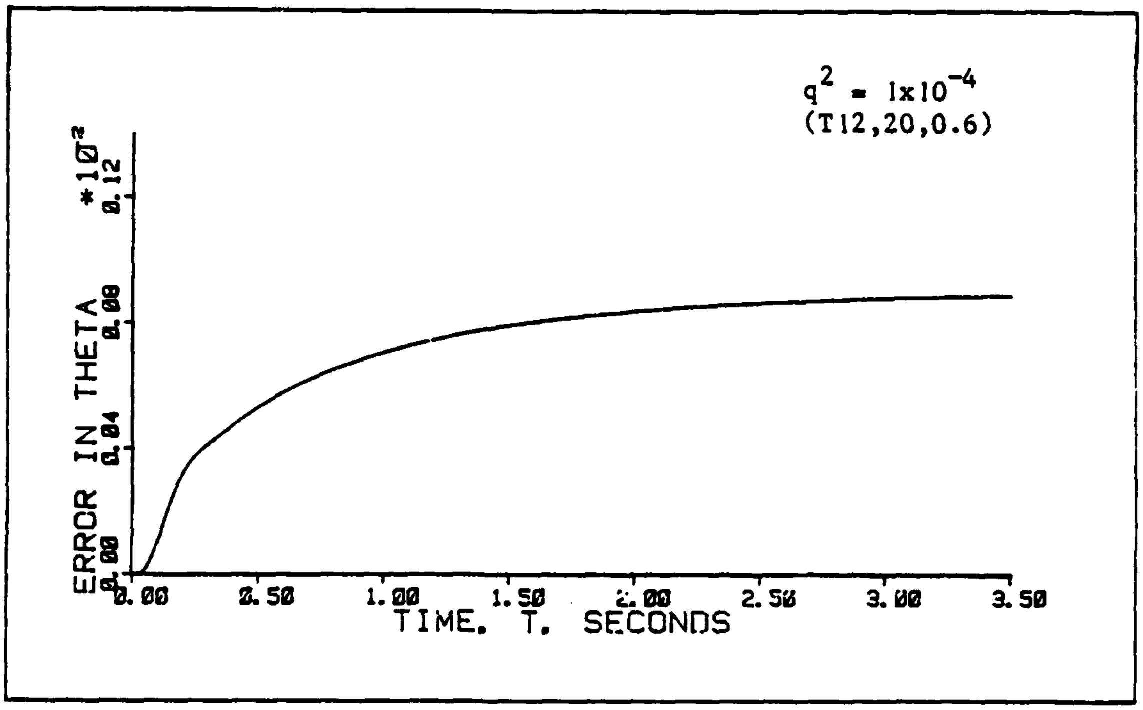


Figure 6-10b: Standard Deviation of θ With White Noise Addition at Off-Design Flight Condition

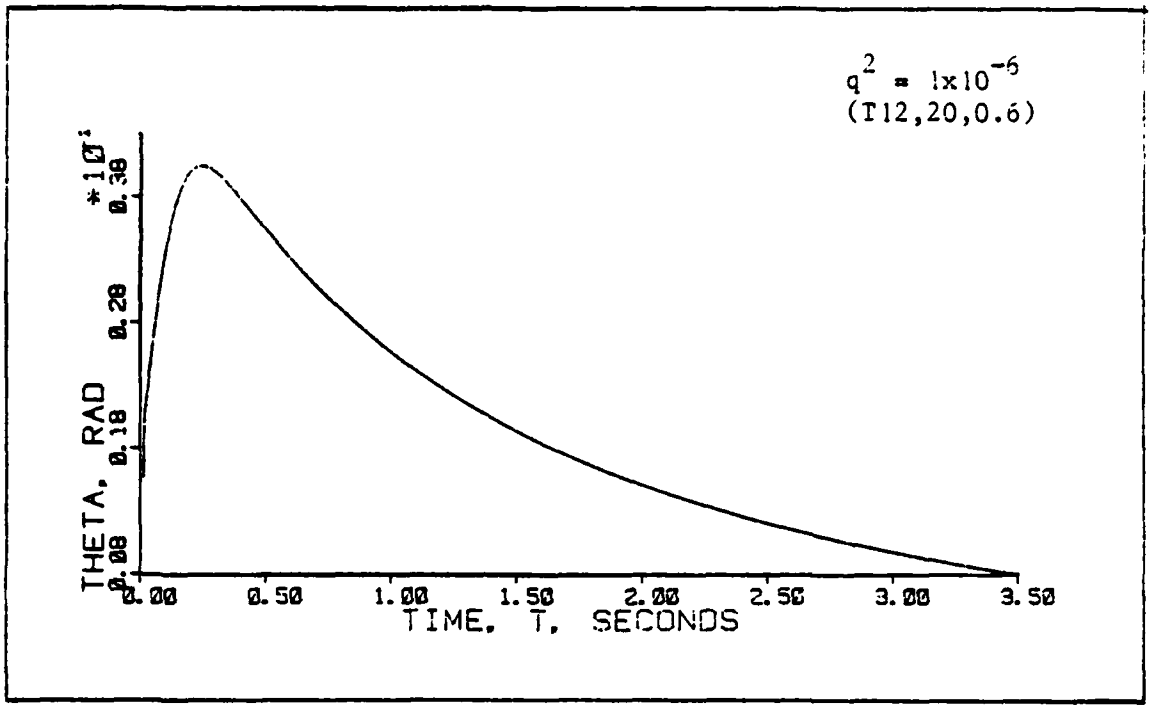


Figure 6-10c: Mean of Θ With White Noise Addition with an Off-Design Flight Condition

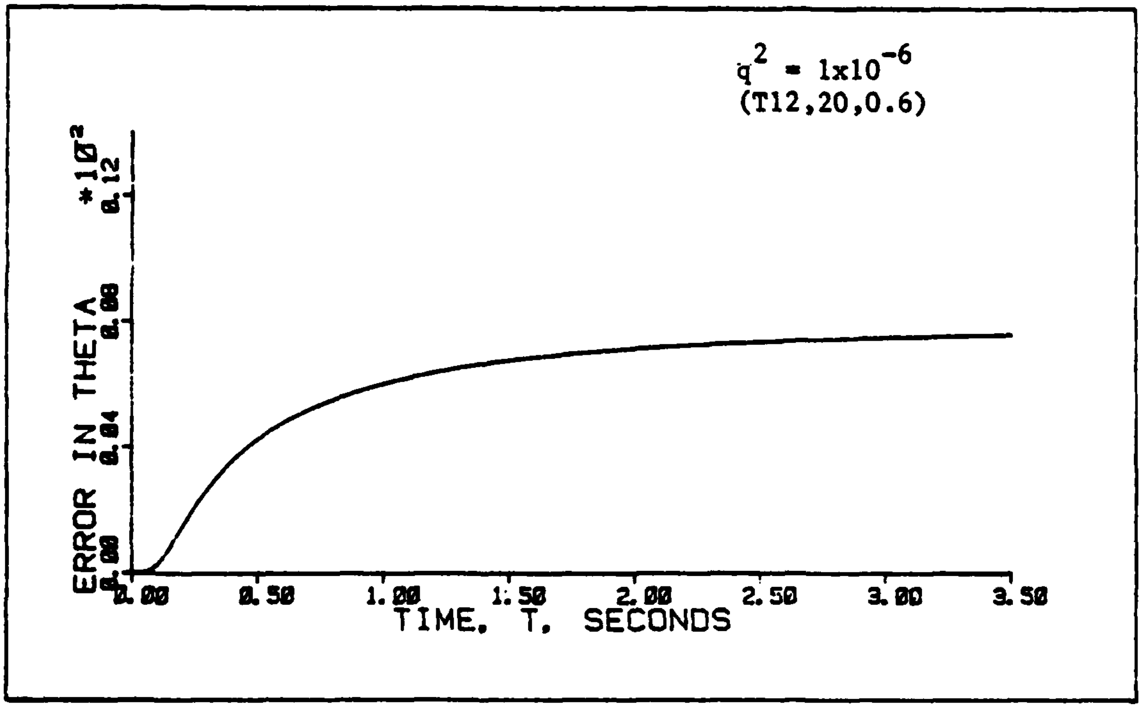


Figure 6-10d: Standard Deviation of Θ With Whise Noise Addition at An Off-Design Flight Condition

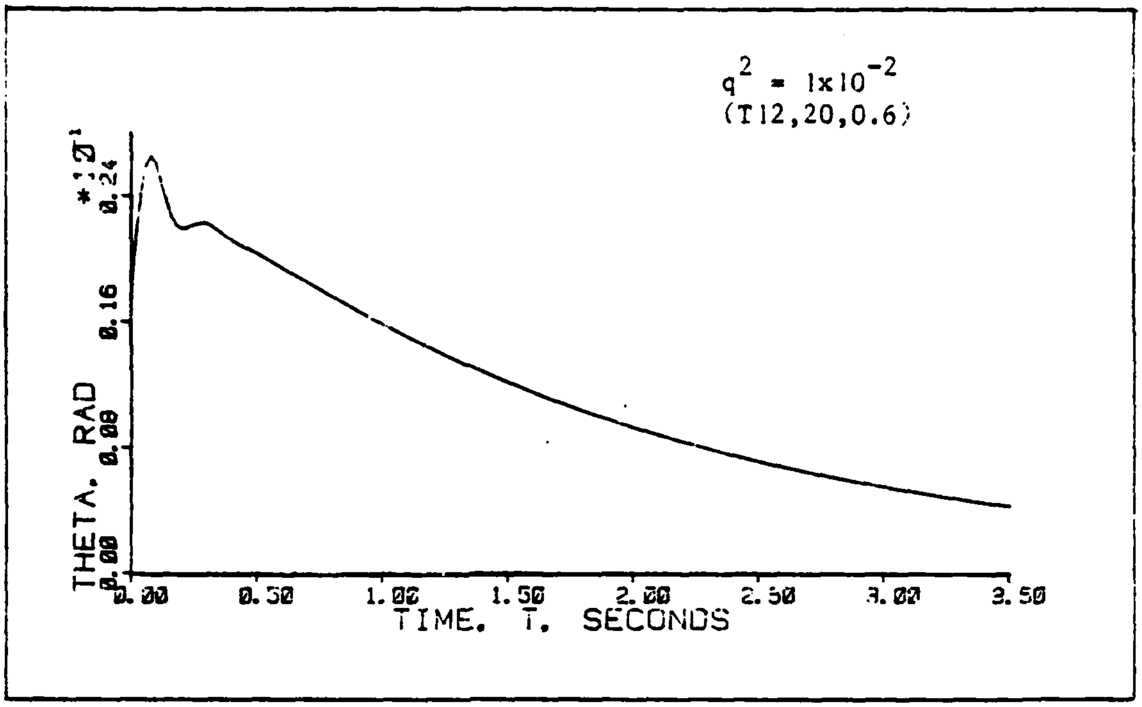


Figure 6-10e: Mean of Θ With White Noise Addition at Off-Design Flight Condition

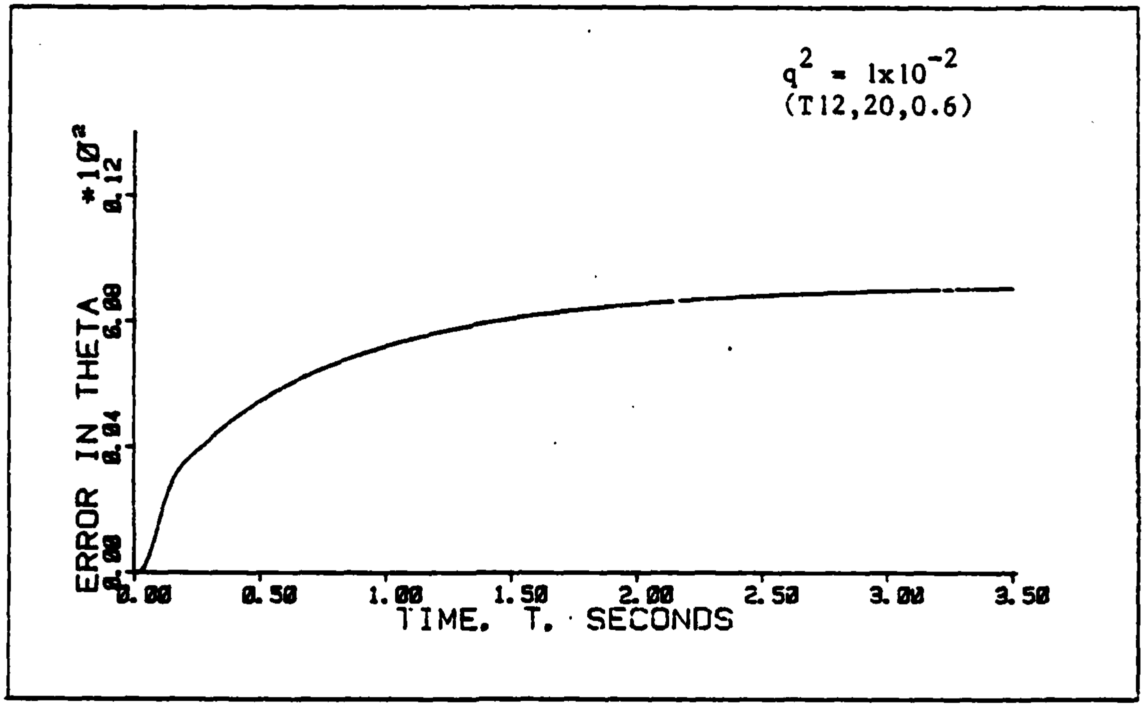


Figure 6-10f: Standard Deviation of Θ With White Noise Addition at Off-Design Flight Condition

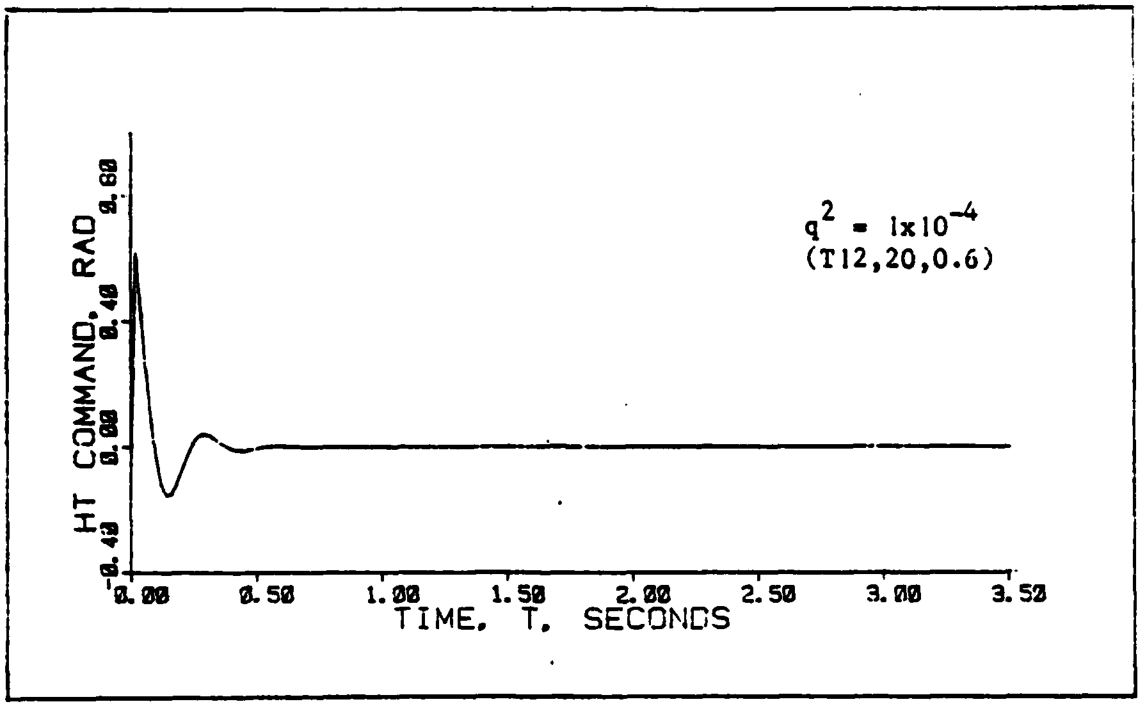


Figure 6-11a: Mean Horizontal Tail Commanded Deflection With White Noise Addition

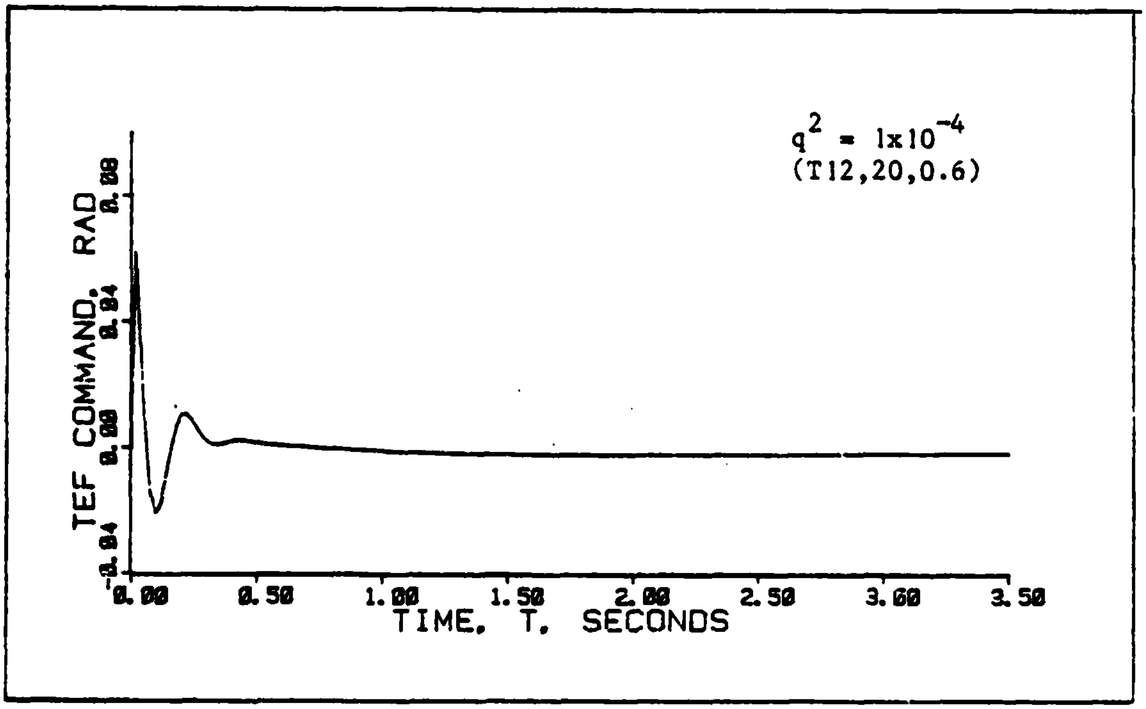


Figure 6-11b: Mean Trailing Edge Flap Commanded Deflection With White Noise Addition

the white noise. This is shown in Figure (6-12). At the off-design flight condition which was initially unstable with no noise addition, the response is stable with the mean of θ approaching zero, and the standard deviation approaching a finite value.

As shown in Figure (6-13), the addition of colored noise generated by a second-order shaping filter with a maximum intensity of $1x10^{-4}$ is not sufficient to stabilize the system. The standard deviation of θ is growing with time, although the mean is approaching zero. However, increasing the intensity to $1x10^{-2}$, the standard deviation does approach a finite value as shown in Figure (6-14).

Thus, Figures (6-10) through (6-14) demonstrate that the stability robustness of the controlled system is enhanced by the addition of all three types of input noise, such that the response of the system is stable even in the face of parameter changes in the real world. White and first-order colored noise of the same maximum intensity produce very similar results. Robustness can also be improved using second-order colored noise, but a greater maximum noise intensity is necessary to achieve a comparable response to the previous two cases. Figure (6-15) shows a power spectral density plot of the second-order time-correlated noise for the case of $Q_u = 2100$. The magnitude of the time-correlated noise at low frequencies (at 0.1 Hz and below) was found to be $1.3x10^{-6}$, after converting from decibels to a linear magnitude scale. Inputting a white noise of this approximate intensity is sufficient to recover the stability of the system as is demonstrated in Figure (6-16). This implies that the robustness improvement is not primarily due to the time-correlated noise concentrated around 70 rad/sec; rather the magnitude of the noise

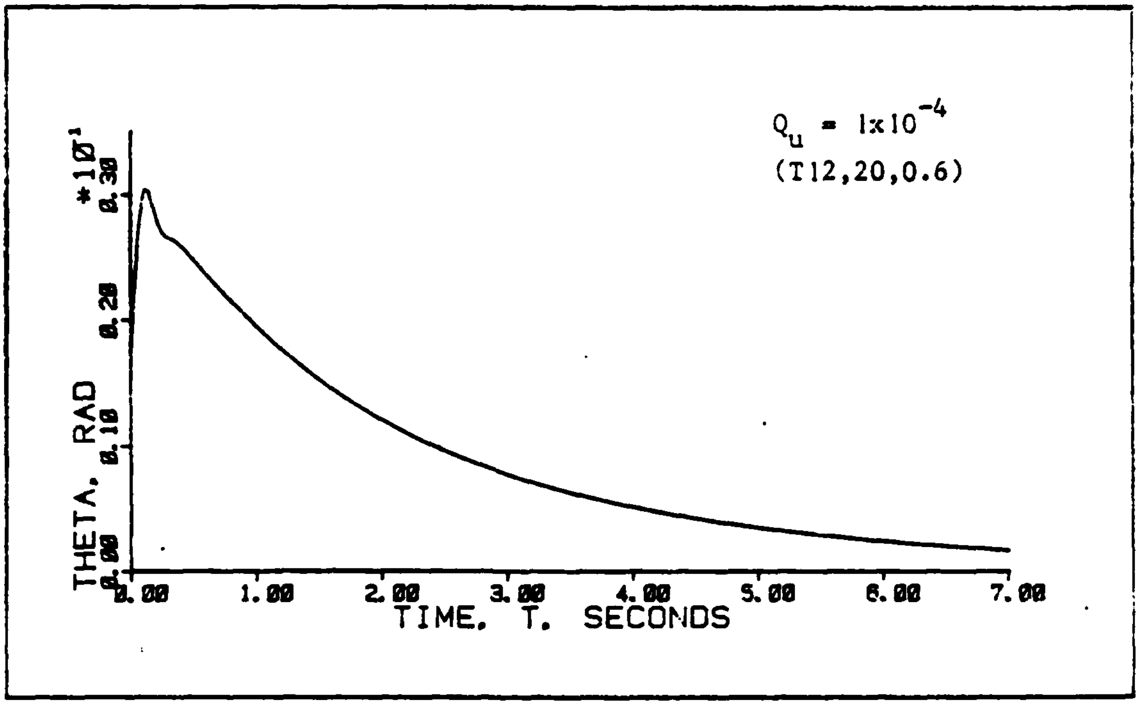


Figure 6-12a: Mean of Θ With First-Order Colored Noise Addition at Off-Design Flight Condition

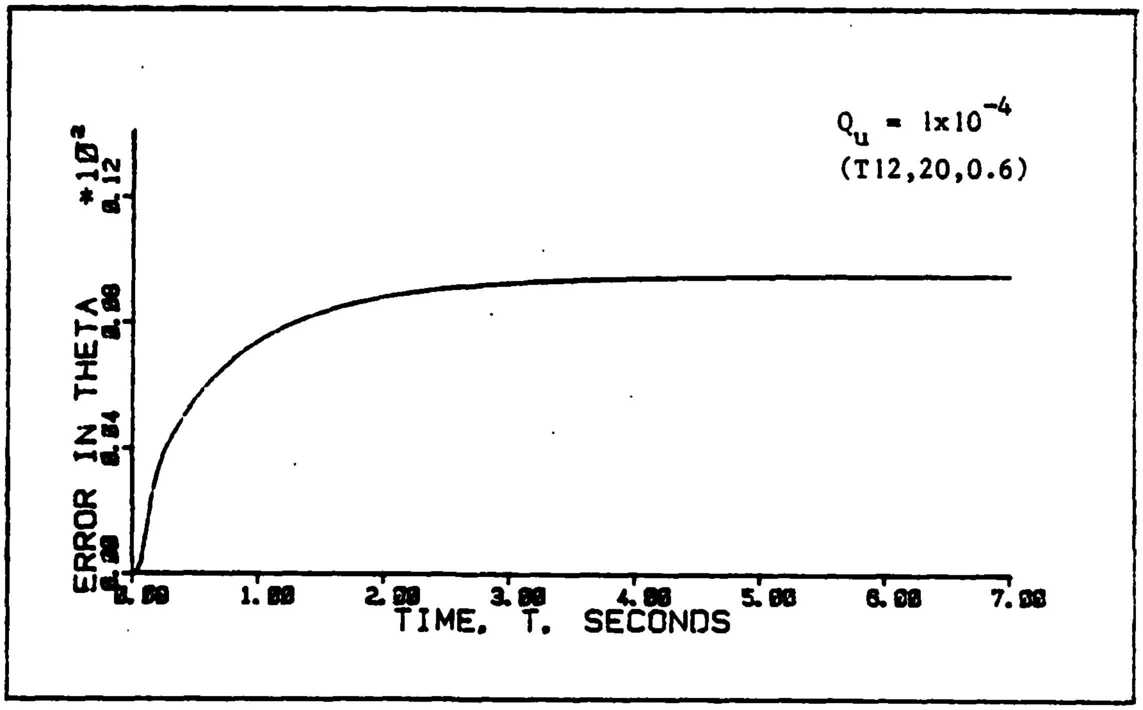


Figure 6-12b: Standard Deviation of Θ With First-Order Colored Noise Addition at Off-Design Flight Condition

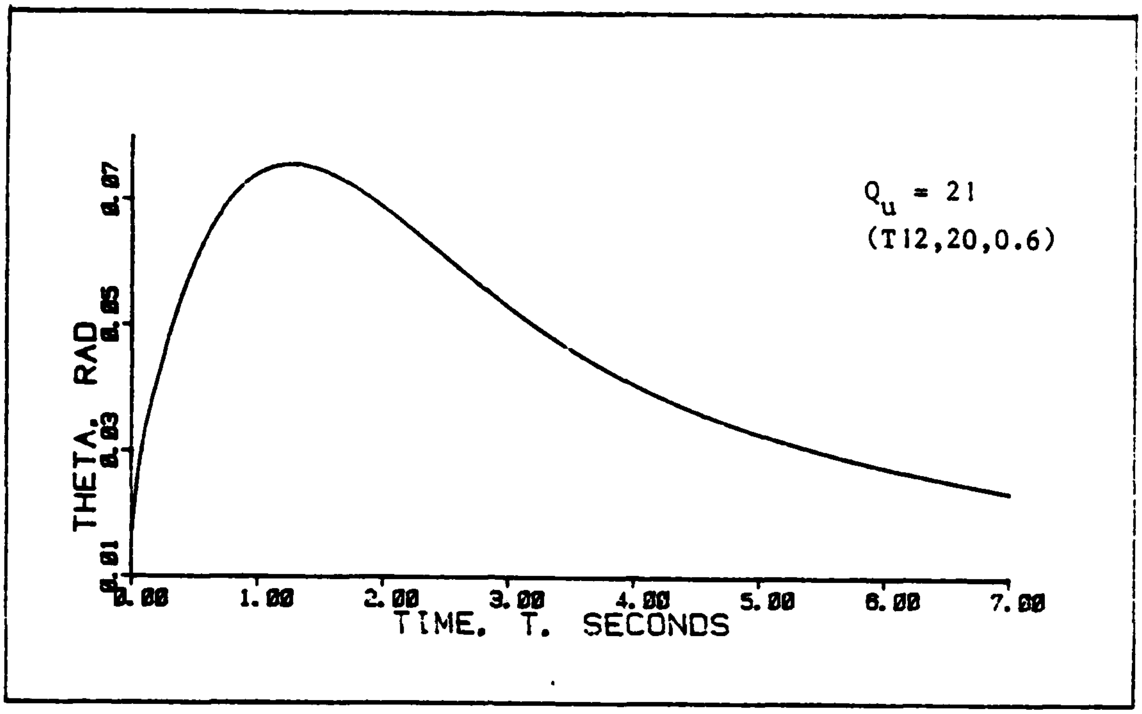


Figure 6-13a: Mean of Θ With the Second-Order Colored Noise Addition at Off-Design Flight Condition

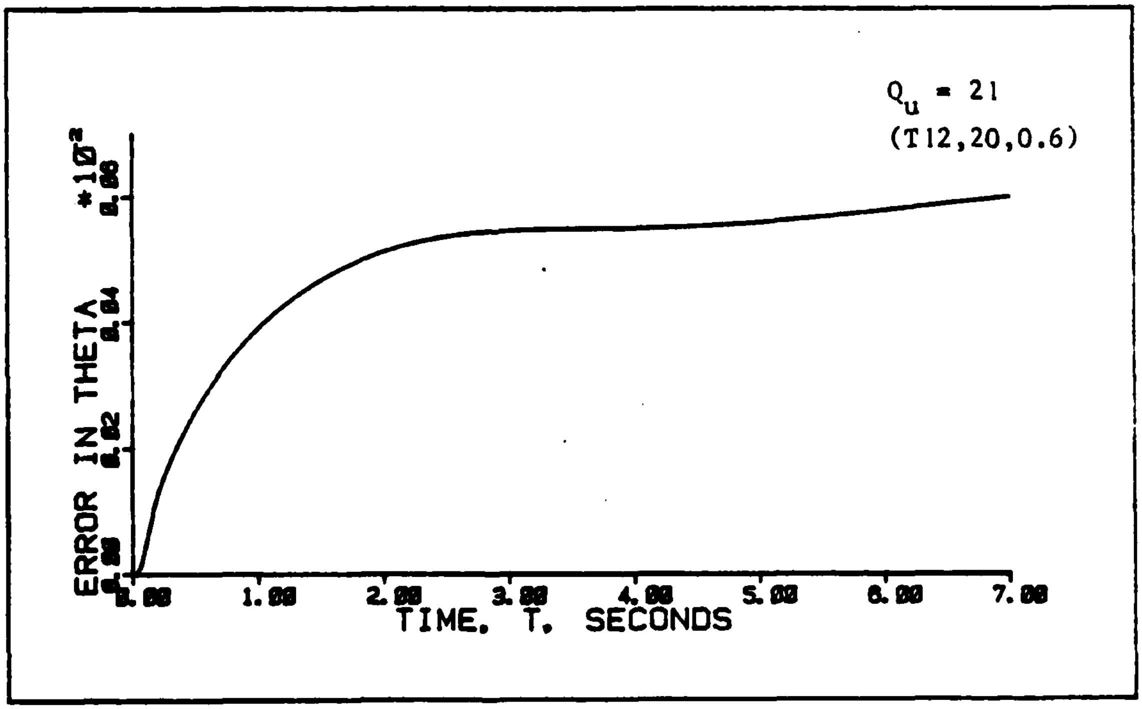


Figure 6-13b: Standard Deviation of Θ With Second-Order Colored Noise Addition at Off-Design Flight Condition

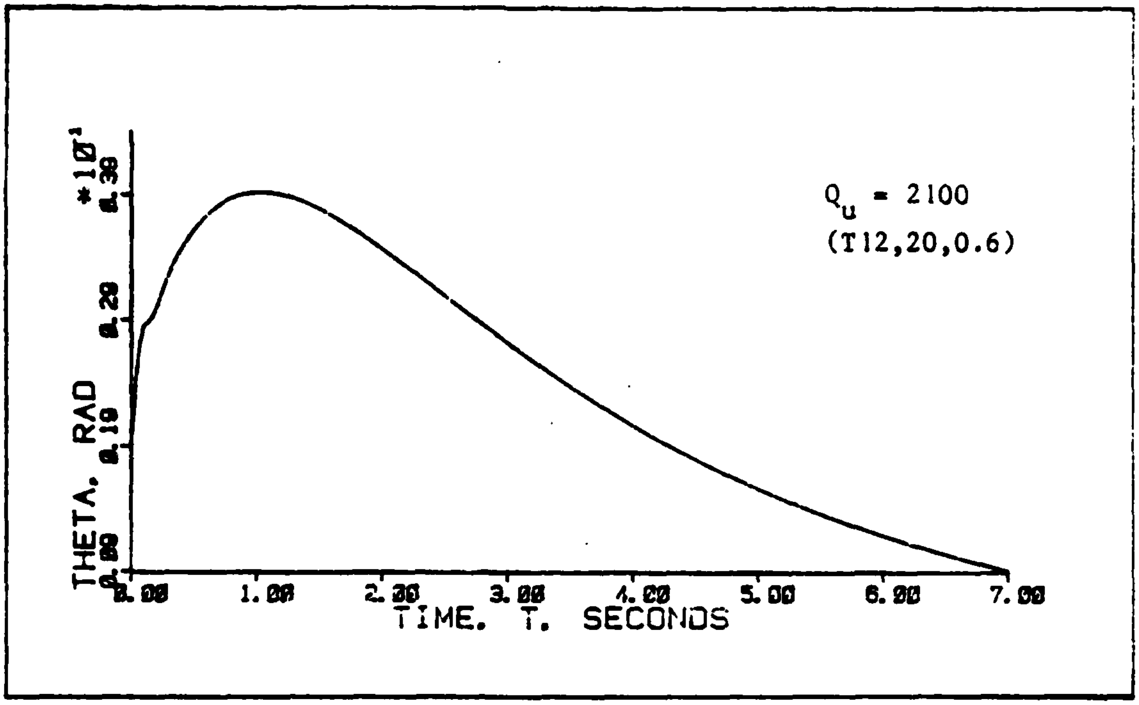


Figure 6-14a: Mean of Θ With Second-Order Colored Noise Addition at Off-Design Flight Condition

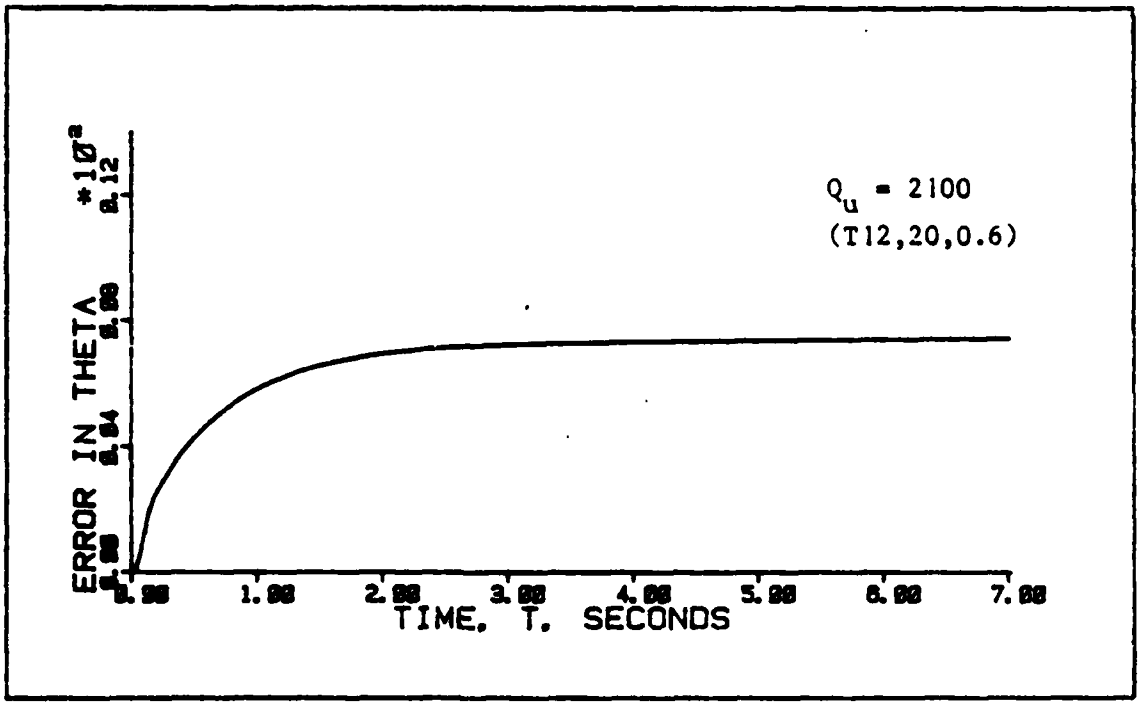


Figure 6-14b: Standard Deviation of Θ With Second-Order Colored Noise Addition at Off-Design Flight Condition

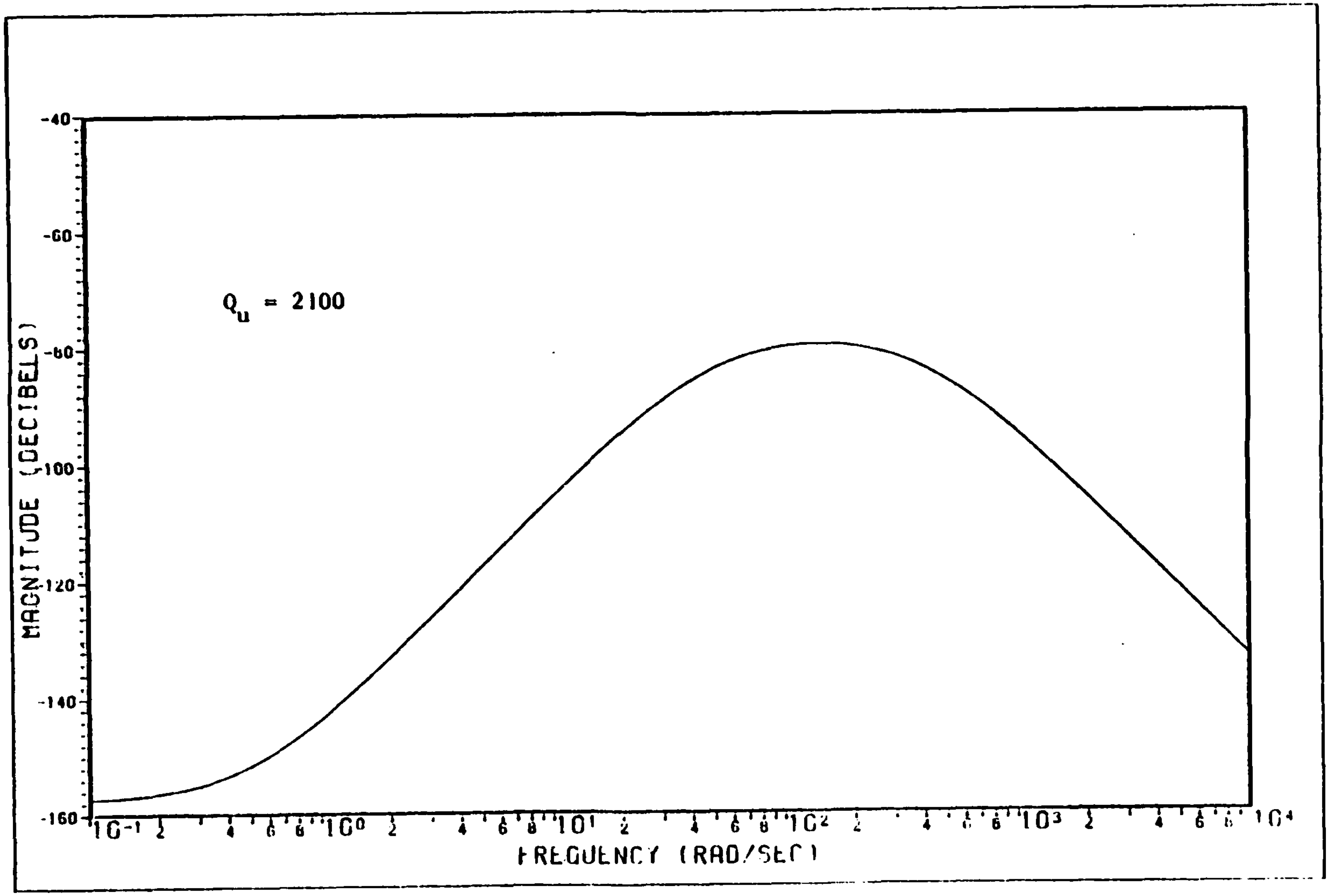


Figure 6-15: Power Spectral Density Function for a Second-Order Colored Noise Process

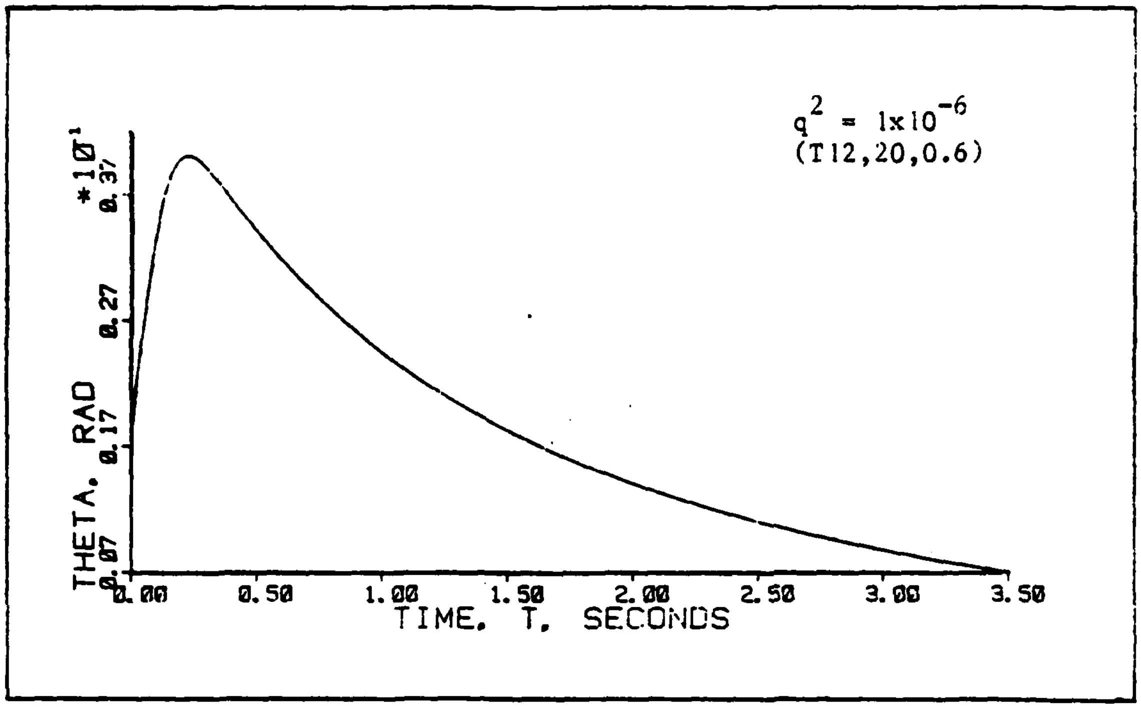


Figure 6-16a: Mean of Θ With White Noise Addition at Off-Design Flight Condition

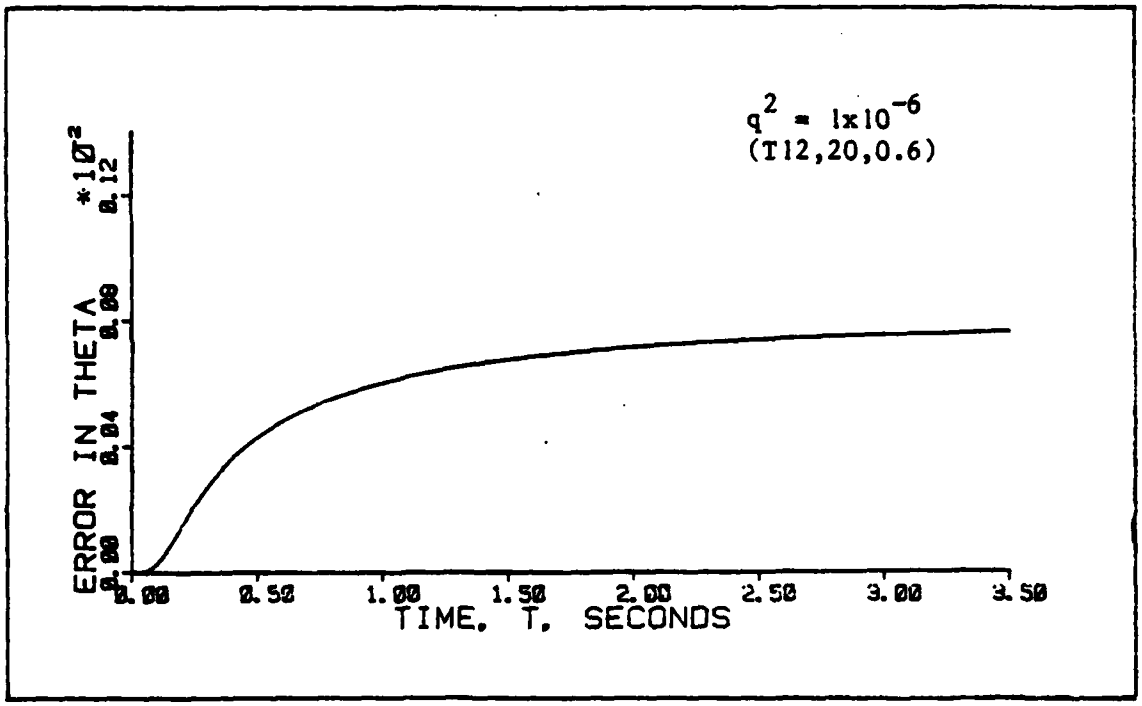


Figure 6-16b: Standard Deviation of Θ With White Noise Addition at Off-Design Flight Condition

elsewhere is sufficiently high to achieve the desired robustification.

6.2.3 Discretized Continuous-Time LQG Regulators at Design Condition

This method of extending the robustification techniques to discrete-time systems was introduced in Section 2.7.1. A continuous-time controller identical to that of Sections 6.2.1 and 6.2.2 is designed, then discrete controller equations are formed by making first-order approximations to the discrete controller gain matrices. The sample rate of the discretized controller is 50 Hertz. The design model matrices are given in Section 6.4.

The time histories of the mean and standard deviation of θ for the eight-state controller evaluted against a truth model of the same dimension are shown in Figure (6-17). Again, the state was given a perfectly known initial condition of one degree. The performance of the discretized controller is very similar to that for the continuous-time case; the state converges to zero fairly slowly. This similarity is as expected since the sample period is short compared to natural system transients. Figure (6-18) shows the response of the same controller evaluated against a twelve-state truth model with higher-order actuator models. As can be seen, changing the environment in which the controller is evaluated introduces a slightly larger initial overshoot and steady-state error in the mean of θ response. Again, the unmodified system does not display good robustness properties.

Figure (6-19) shows the response of the same controller with the filter robustified by adding a white noise of strength $q = 1\text{x}10^{-6}$ to the design model. The improvement in transient time, initial overshoot, and

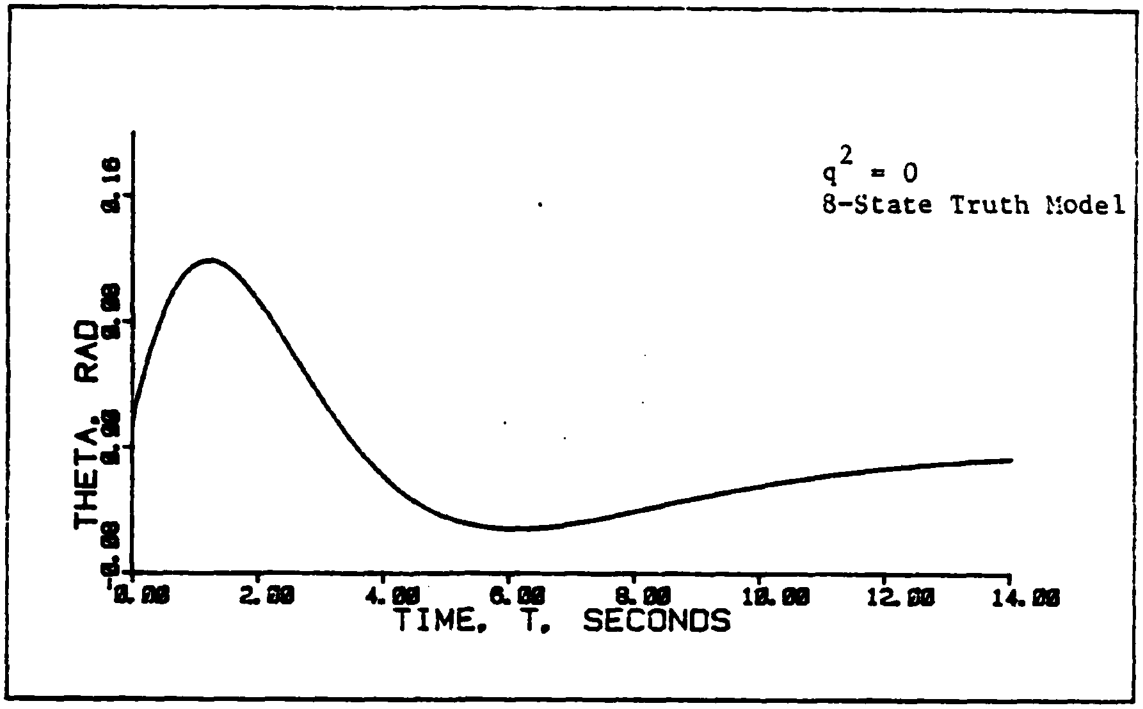


Figure 6-17a: Mean of Θ at the Design Condition

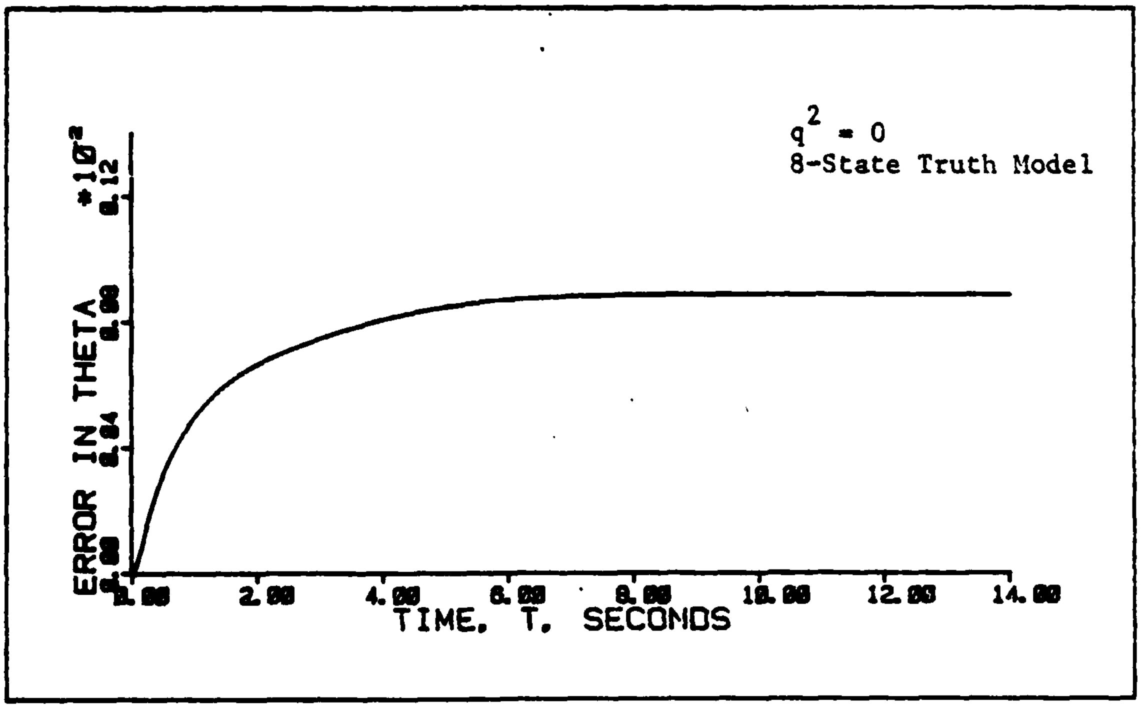


Figure 6-17b: Standard Deviation of Θ at the Design Condition

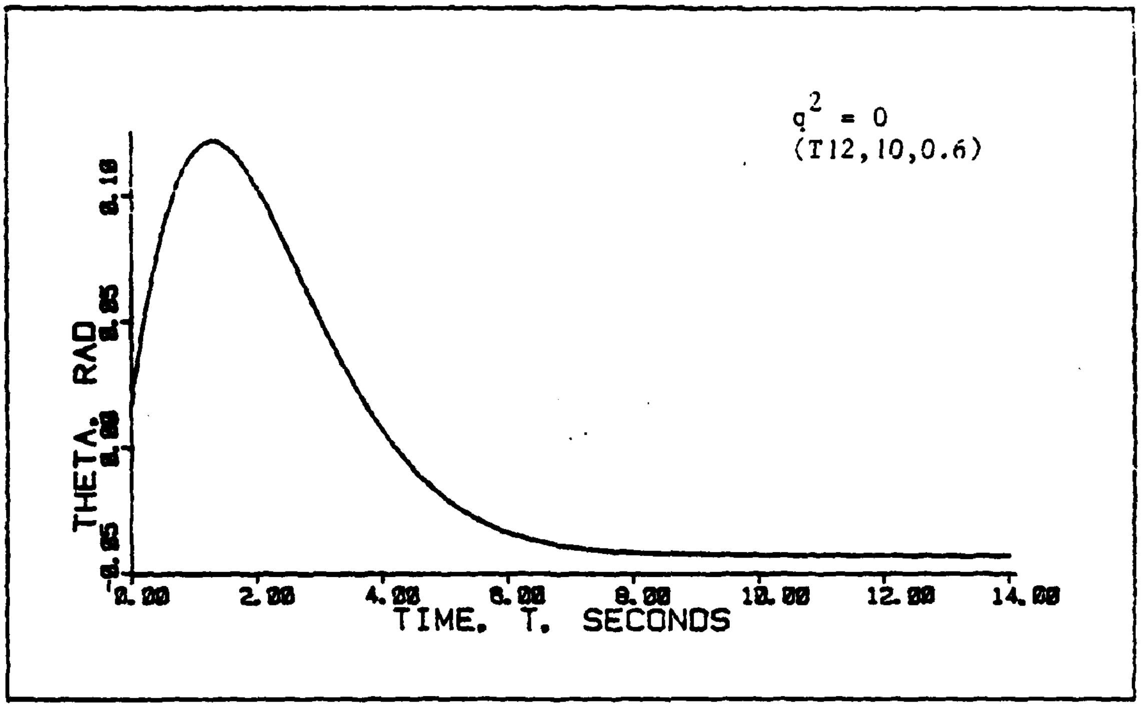


Figure 6-18a: Mean of Θ With No Noise Addition

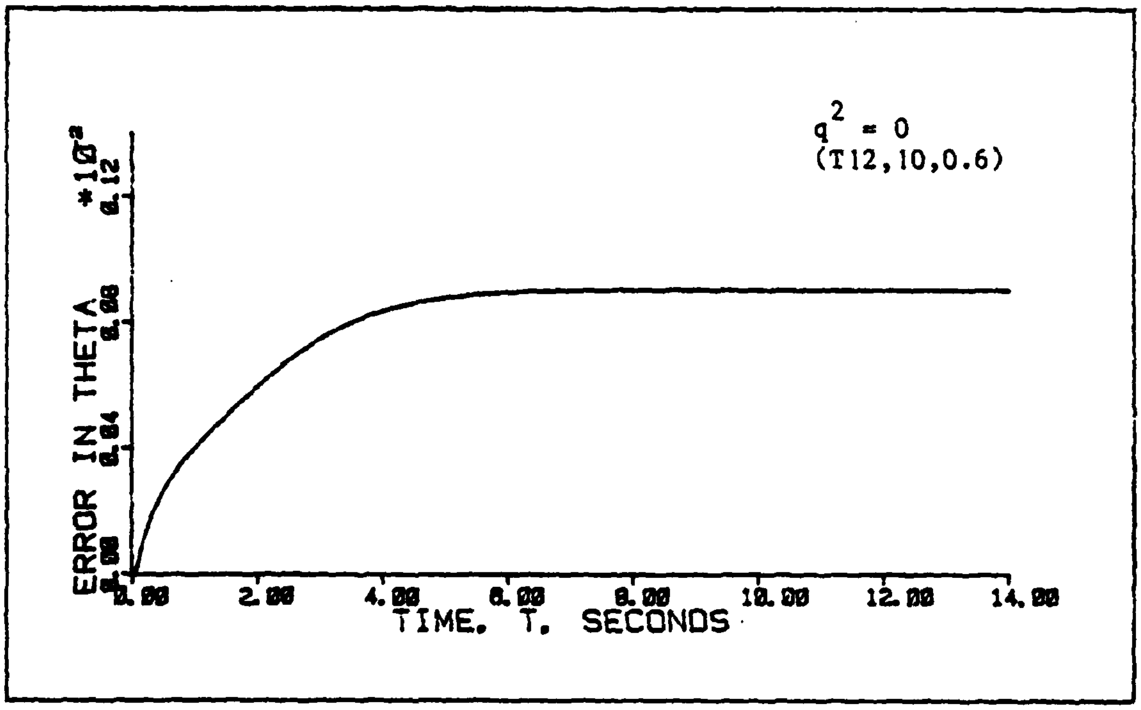


Figure 6-18b: Standard Deviation of Θ With No Noise Addition

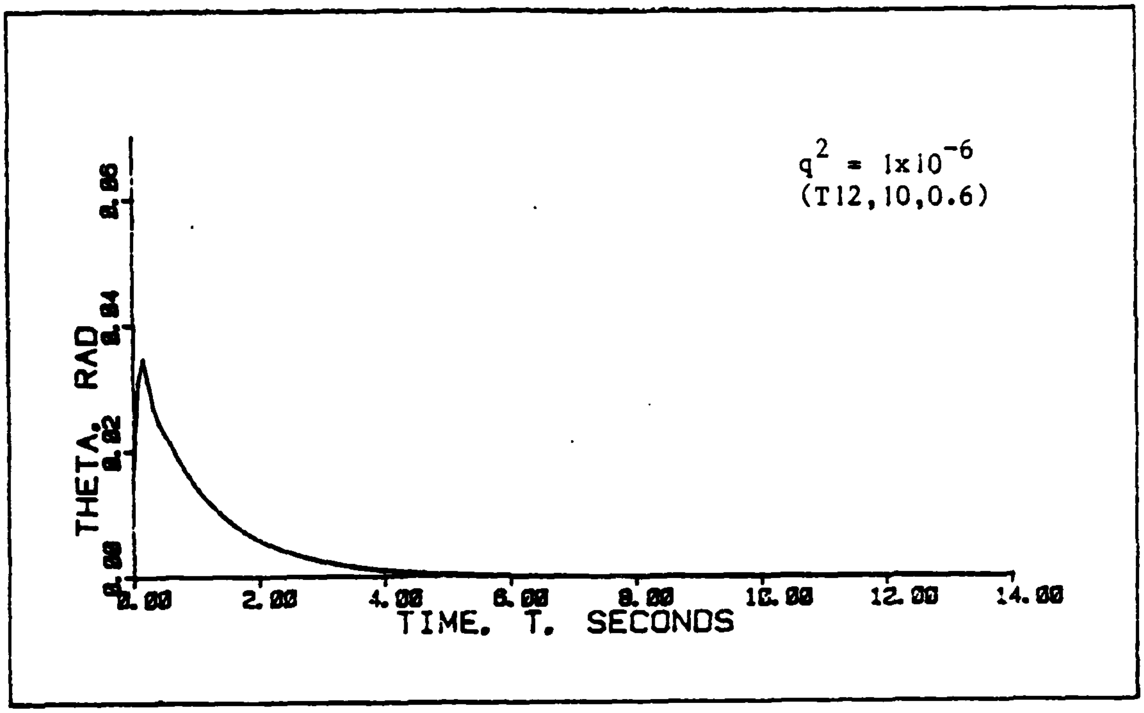


Figure 6-19a: Mean of Θ With White Noise Addition

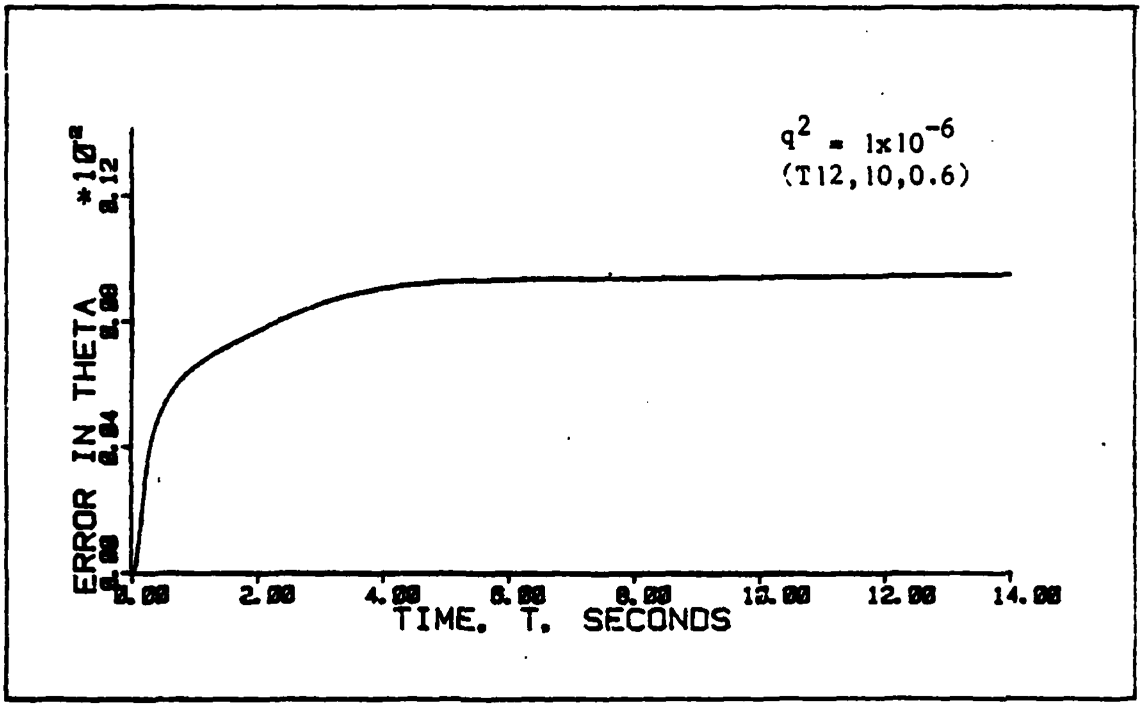


Figure 6-19b: Standard Deviation of Θ With White Noise Addition

steady-state value is comparable to the continuous-time case with no noticeable change in the standard deviation.

The results of applying colored noise generated by a first-order shaping filter are shown in Figure (6-20). The overshoot is larger than for the white noise addition, and there is still a slight steady-state mean error, but the improvement over the unrobustified case is still substantial.

It was found that augmenting second-order shaping filter states with the design model, designing a continuous-time controller, than discretizing the concroller produced an unstable closed-loop system for any value of Q_u. This characteristic was not observed in the continuous-time case.

In addition, as described previously in Reference 21, it was observed that the strength of the white and colored noise could not be adjusted arbitrarily upwards. It was found that, with the twelve-state truth model (T12,10,0.6), q^2 and Q_u could be adjusted between zero and $1x10^{-6}$, and the robustification improvement was similar to the continous-time case. However, noise intensities beyond this value ($Q_u = 2x10^{-6}$) would actually drive closed-loop system eigenvalues outside of the z-domain unit circle.

A comparison of the steady-state values of the standard deviations of the aircraft states are given in Table (6-4). The table includes values for a system with no noise addition, white and first-order colored noise of the same maximum intensity. The results are similar to the continuous-time case. It is seen that the addition of white noise increases the values of the standard deviations of the states. As in the previous case, colored noise addition generated by a first-order shaping filter did not yield lower standard deviations than white noise

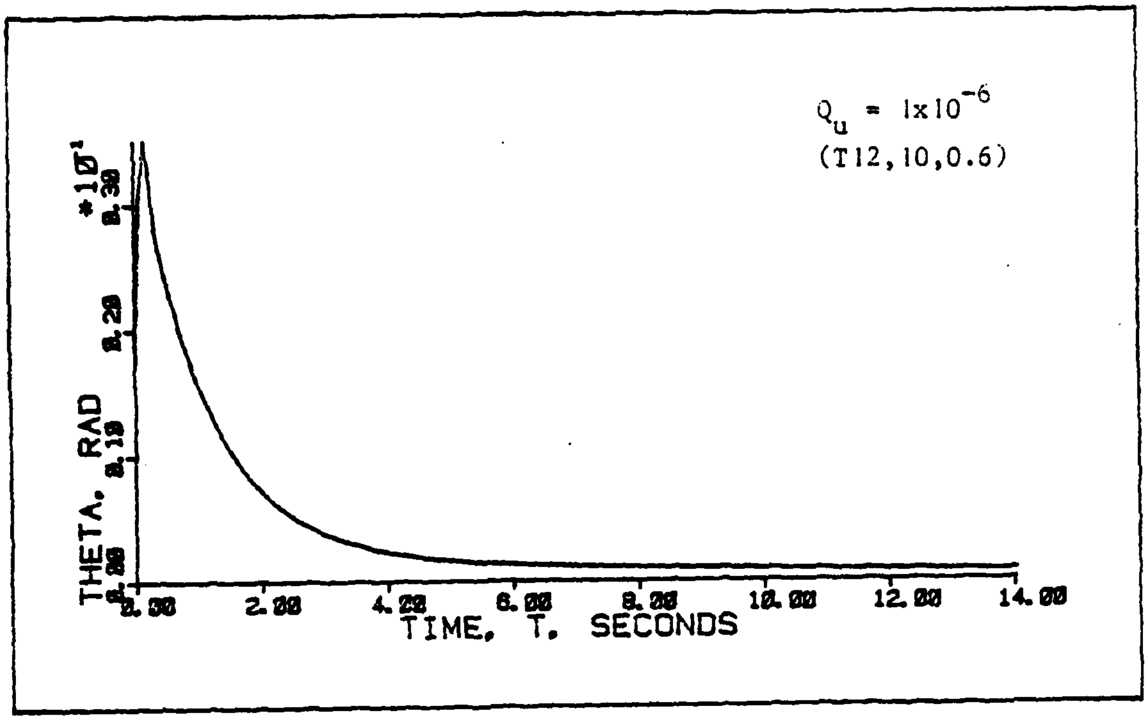


Figure 6-20a: Mean of Θ With First-Order Colored Noise Addition

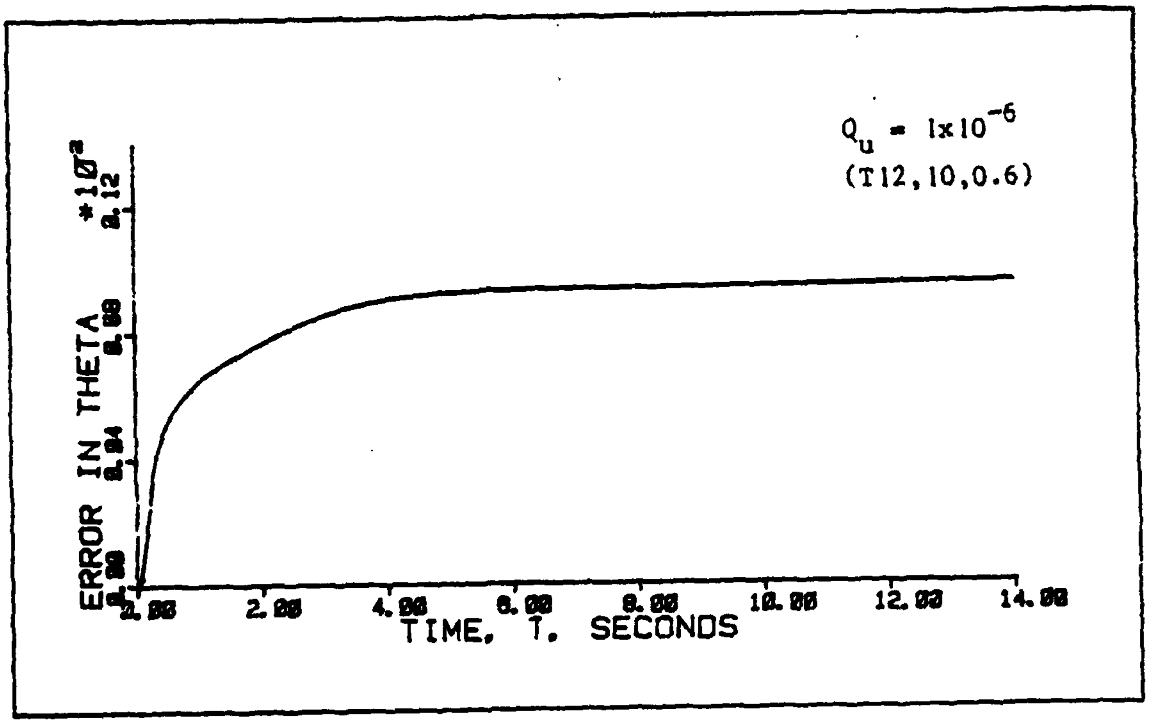


Figure 6-20b: Standard Deviation of Θ With First-Order Colored Noise Addition

Table 6-4

Comparison of Steady-State Standard Deviations of Aircraft States at the Design Condition for a Discretized Continuous-Time System

	q^2	Q_u	σ_Θ	σ_α	σ_q
No Noise	0	-	8.877×10^{-4}	5.035×10^{-3}	1.258×10^{-3}
White Noise	1×10^{-4}	-	9.645×10^{-4}	5.310×10^{-3}	2.113×10^{-3}
1st Order Shaping Filter	-	1×10^{-4}	1.005×10^{-3}	5.821×10^{-3}	1.817×10^{-4}
2nd Order Shaping Filter	-	-	Unstable	Unstable	Unstable

Table 6-5

Comparison of Steady-State Standard Deviations of Aircraft States With Higher-Order Actuators For a Discretized Continuous-Time System

	q^2	Q_u	σ_Θ	σ_α	σ_q
No Noise	0	-	9.053×10^{-4}	4.899×10^{-3}	1.257×10^{-3}
White Noise	1×10^{-6}	-	9.511×10^{-4}	5.198×10^{-4}	2.825×10^{-4}
1st Order Shaping Filter	-	1×10^{-6}	9.524×10^{-4}	5.212×10^{-3}	2.758×10^{-4}
2nd Order Shaping Filter	-	-	Unstable	Unstable	Unstable

addition except for the pitch rate, q. Table (6-5) indicates identical trends for the case where higher-order dynamics are included in the truth model. The lower maximum noise intensity than that used for Table (6-4) reflects the fact that, with a higher-dimension truth model, the closed-loop system is driven unstable for a lower value of q^2.

Thus, it is seen that the desired robustness enhancement can be gained by adding white input noise to the system model. Time-correlated noise is not appropriate for this problem because it does not yield performance benefits and only serves to increase the complexity of the design model.

6.2.4 Discretized Continuous-Time LQG Regulators at Off-Design Condition

Figure (6-21) demonstrates that the system exhibits an unstable response when the flight condition is changed to an altitude of 20000 feet and a Mach number of 0.6 (T12,20,0.6).

Again, it was found that the robustness characteristics could be improved with the injection of white and colored noise for a finite range of q^2 and Q_u. At the off-design condition, for values of q^2 and Q_u beyond $2.5\text{x}10^{-5}$, the closed-loop system was unstable.

The response of the system with the maximum value of q^2 is shown in Figures (6-22a) and (6-22b). As can be seen, the stability has been recovered, indicating that the full-state feedback system was stable. Thus, by the addition of white input noise, the characteristics approach that of a full-state feedback controller. Figures (6-22c) and (6-22d) show that stability is recovered with a lower value of q^2. Increasing it beyond this value changes only the transients of the mean and the magnitude of the standard deviation.

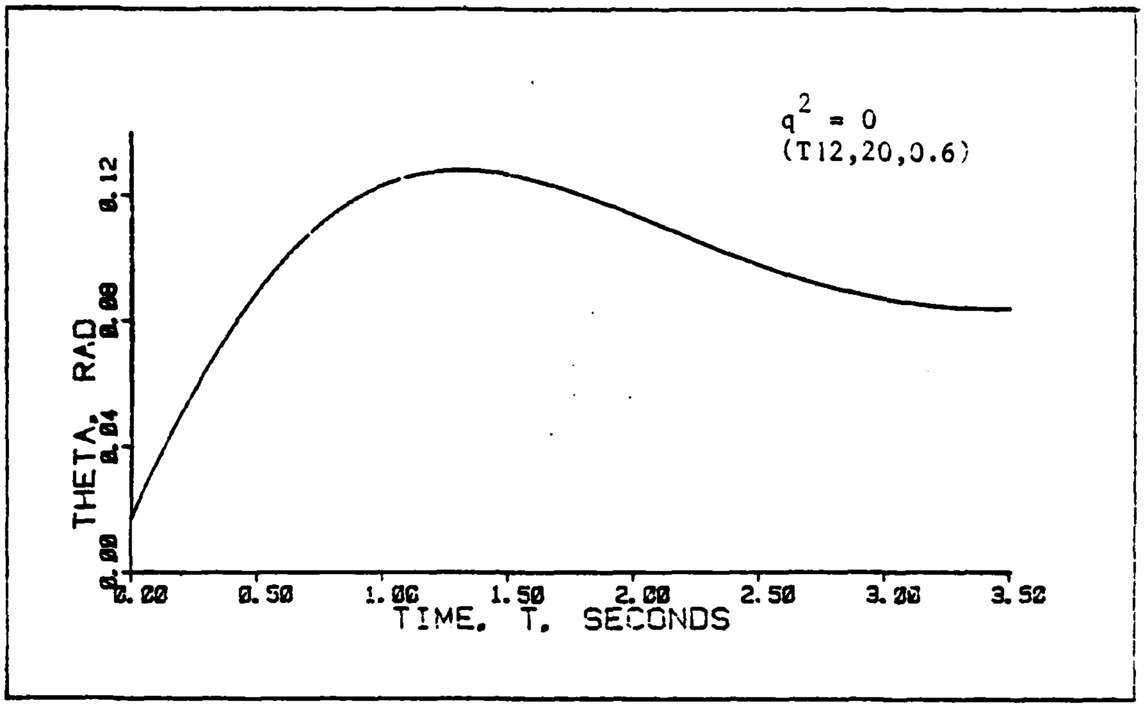


Figure 6-21a: Mean of Θ With No Noise Addition at Off-Design Flight Condition

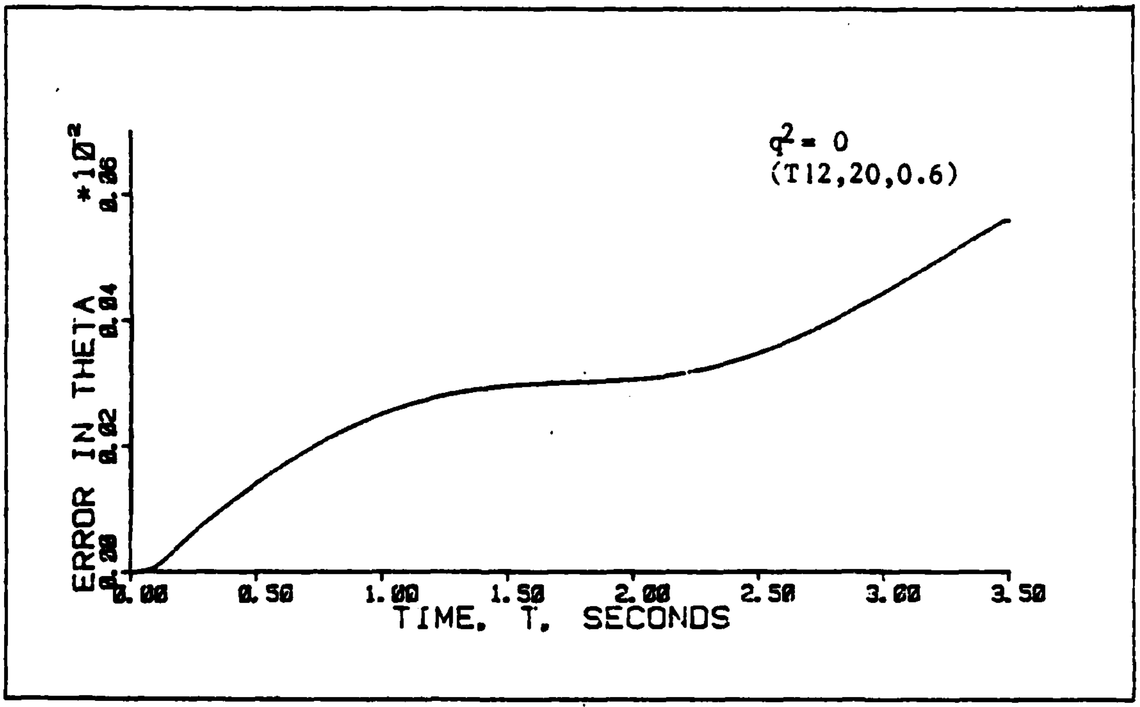


Figure 6-21b: Standard Deviation of Θ With No Noise Addition at the Off-Desi?n Flight Condition

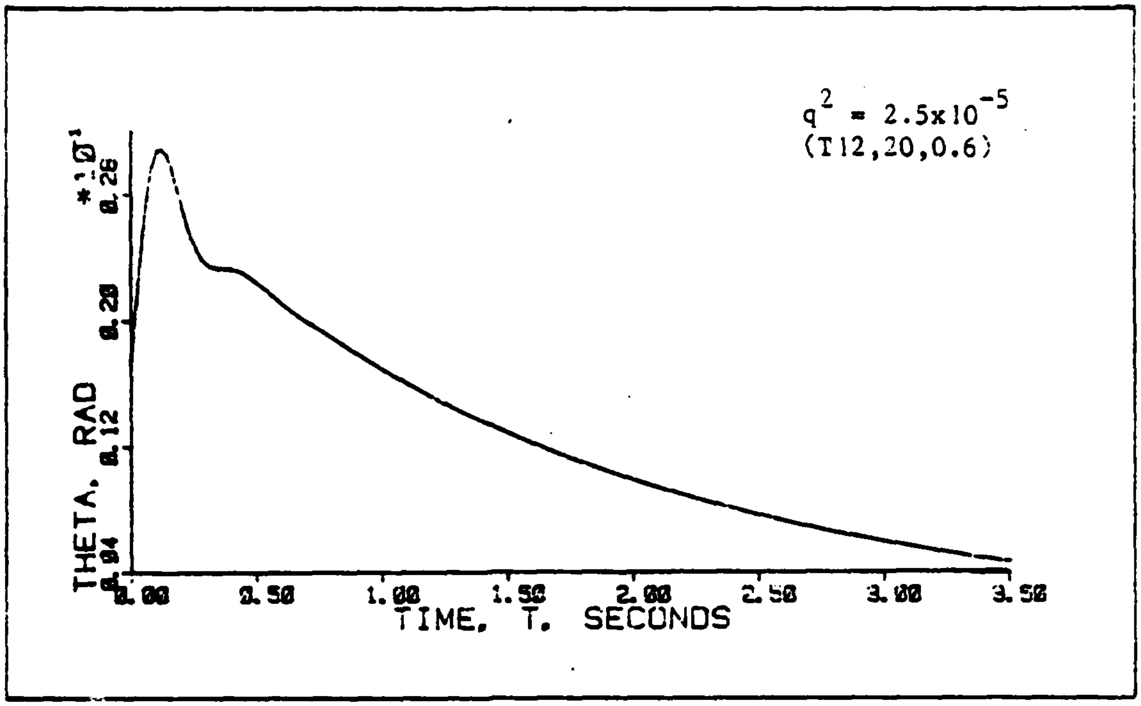


Figure 6-22a: Mean of Θ With White Noise Addition at Off-Design Flight Condition

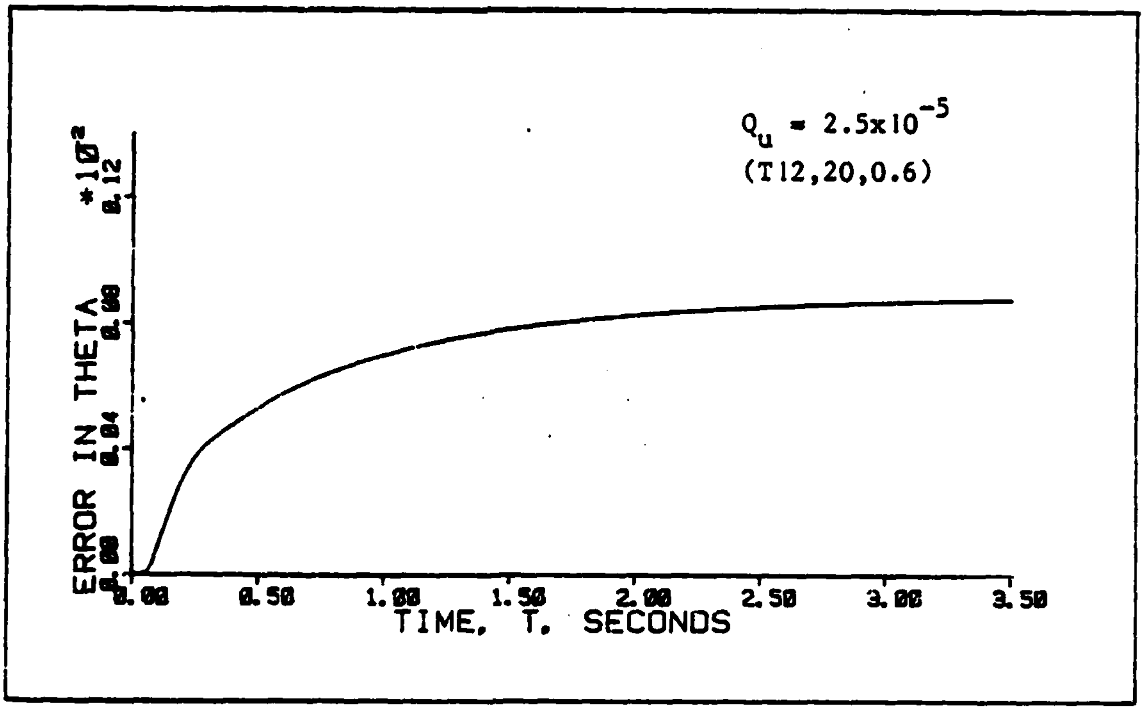


Figure 6-22b: Standard Deviation of Θ With White Noise Addition at the Off-Design Flight Condition

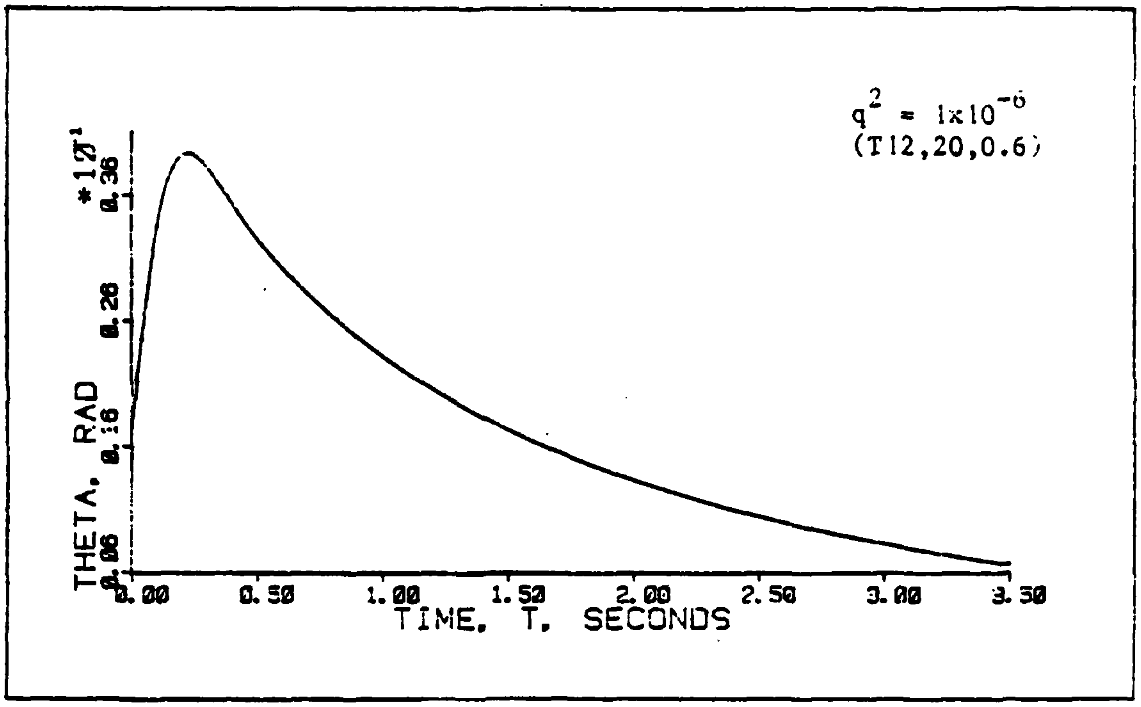


Figure 6-22c: Mean of Θ With White Noise Addition at an Off-Design Flight Condition

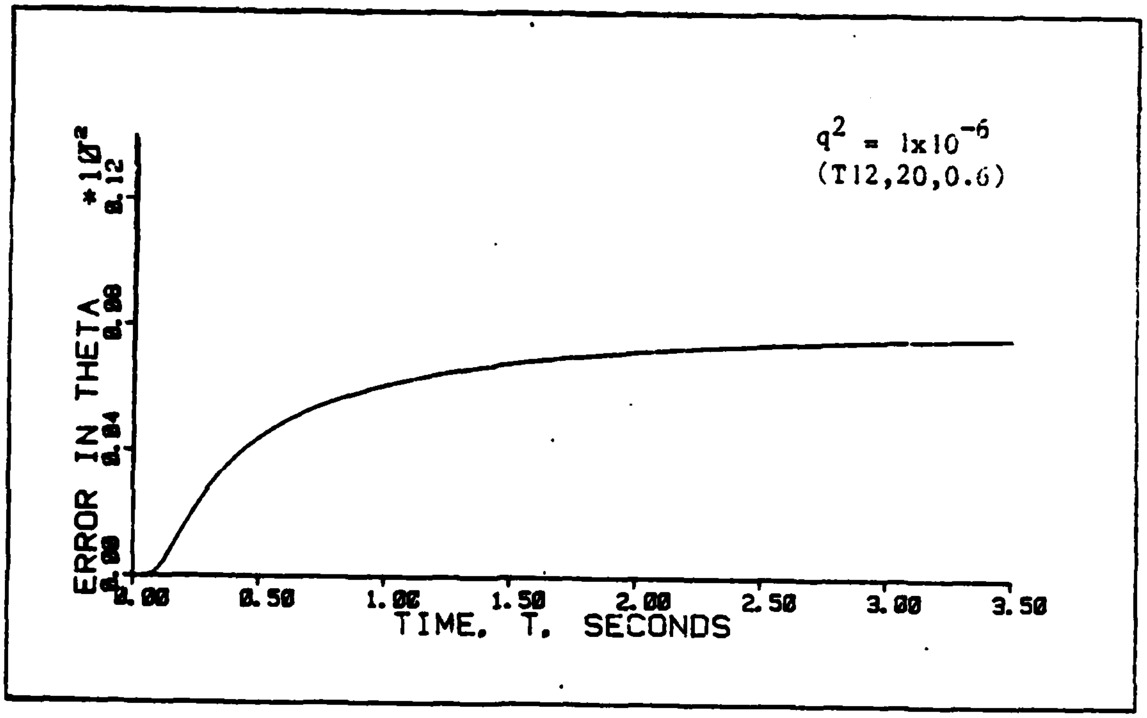


Figure 6-22d: Standard Deviation of Θ With White Noise Addition at an Off-Design Flight Condition

Figure (6-23) demonstrates that adding colored noise generated by a first-order shaping filter achieves the same degree of robustness enhancement with only an increase in the initial overshoot of θ compared to that of Figure (6-22a). The steady-state standard deviations of Figures (6-22b) and (6-23b) are not significantly different.

Augmenting second-order shaping filter states with the system model again resulted in an unstable closed-loop system. For any non-zero value of Q_u, some of the closed-loop system eigenvalues were driven outside the unit circle in the z-domain.

Because of the sensitivity of the system to slight changes in the strength of the added noise, it is felt that this method is inappropriate for this problem. The range of admissible values for q^2 and Q_u is very slight, and the loss of closed-loop system stability is very abrupt when the range is exceeded. In addition, there is no range of Q_u which improves the robustness characteristics of the controlled system when colored-noise generated by a second-order filter is added to the model.

6.2.5 Sampled-Data LQG Regulators at Design Condition

This method of extending the robustification techniques to sampled-data systems was introduced in Section 2.7.2. For this case, the continuous-time system equations are discretized, then a sampled-data Kalman filter and controller are designed from the onset.

The results of performance analyses for the sampled-data controller with a sample rate of 50 Hertz are found to be extremely similar to those for the continuous-time controller.

Figure (6-24) shows the response of θ to an initial condition of one degree. This is the result of a performance analysis where the dimensions

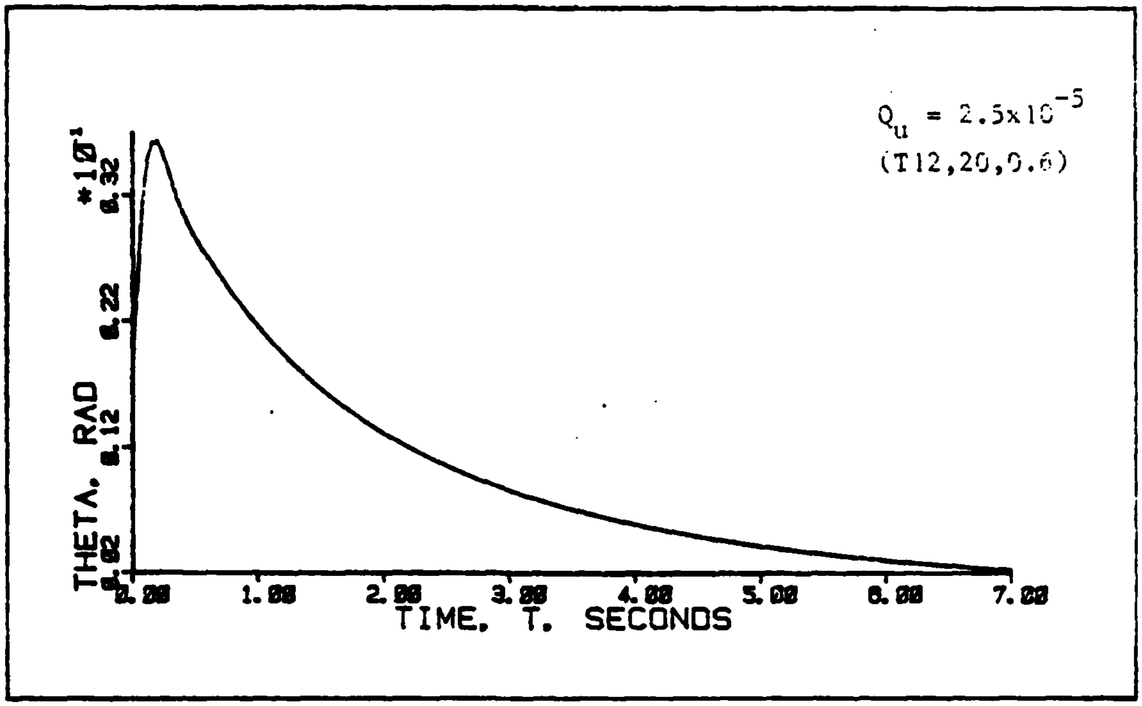


Figure 6-23a: Mean of Θ With First-Order Colored Noise Addition at Off-Design Flight Condition

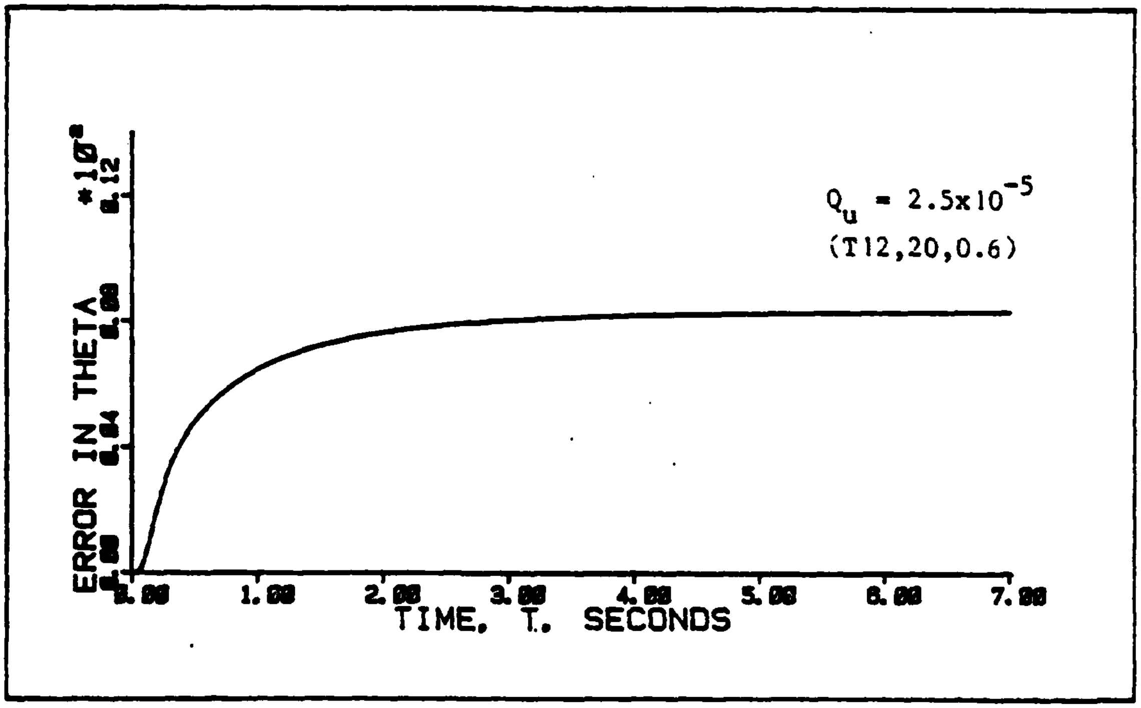


Figure 6-23b: Standard Deviation of Θ With First-Order Colored Noise Addition at an Off-Design Flight Condition

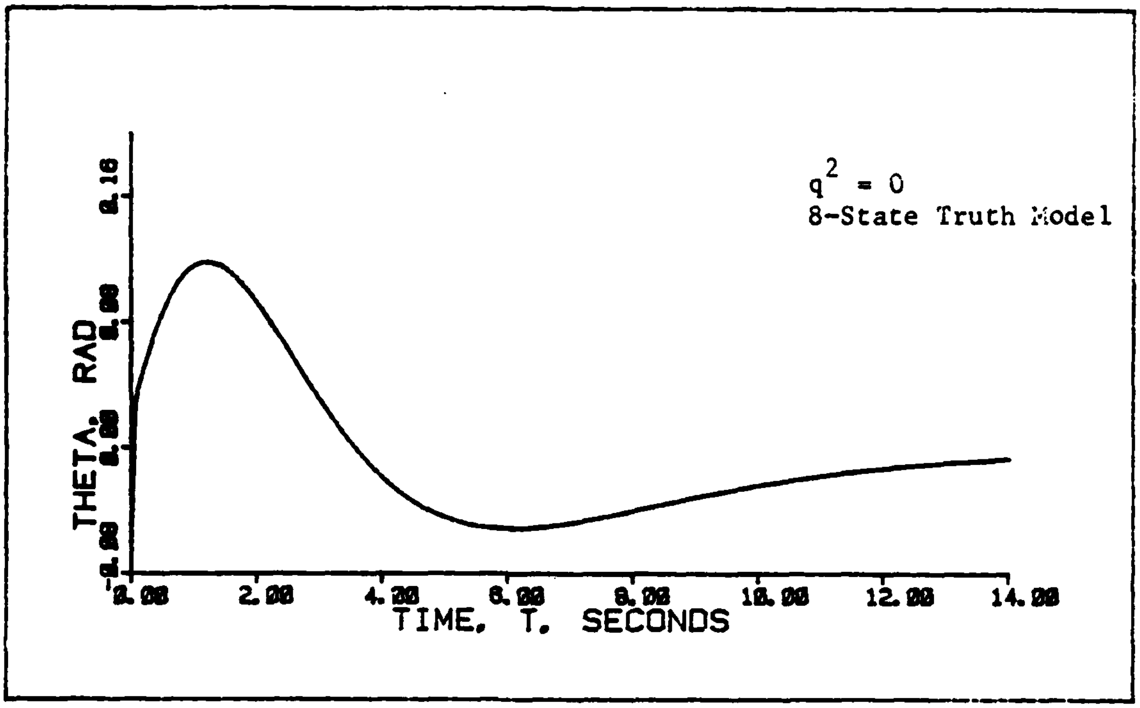


Figure 6-24a: Mean of Θ at the Design Condition

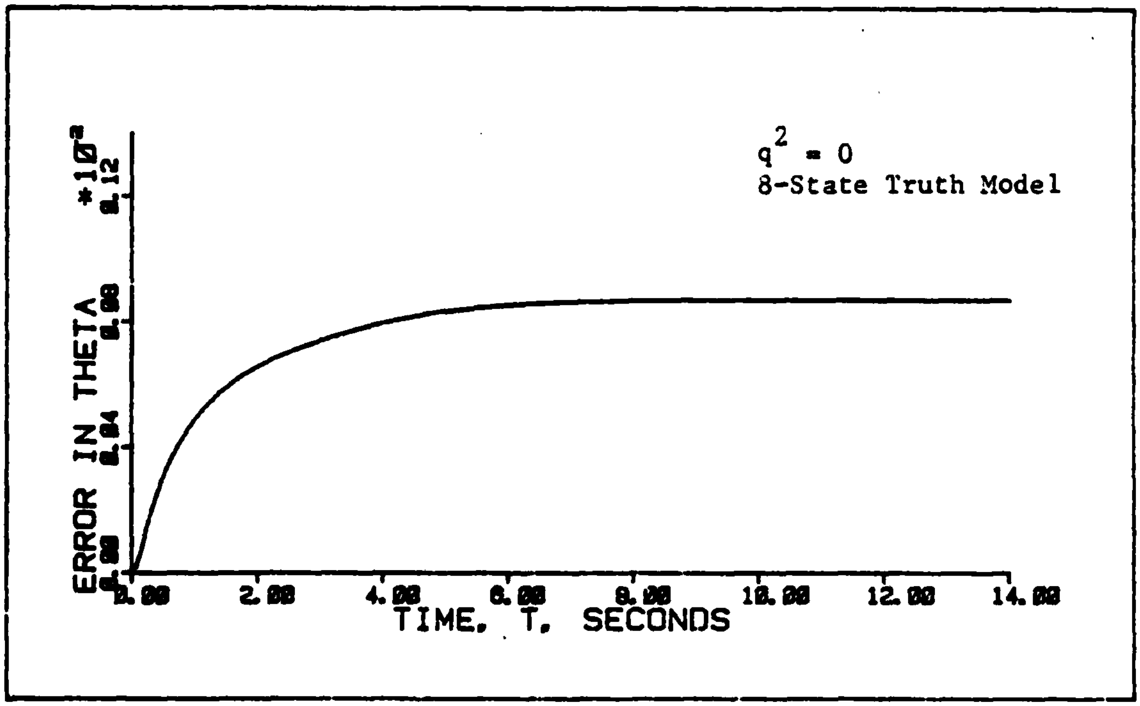


Figure 6-24b: Standard Deviation of Θ at the Design Condition

of the truth model and the design model are the same (Section 5.4). As observed previously, the mean of Θ converges slowly to zero. Figure (6-25) shows the response of the same controller when higher-order actuator dynamics are introduced in the truth model. Again, large initial overshoots and steady-state mean errors are introduced. The unmodified sampled-data controller does not exhibit good robustness properties when certain states are ignored in the design model.

With the addition of white input noise, (as described in Section 2.7.2), the undesirable characteristics of the unrobustified system do not appear in the time response. This is shown in Figures (6-26a) and (6-26b). Figures (6-26c,d,e,f) show the trend for a higher and lower value of q^2. Increasing the magnitude of the white noise changes the transient mean response and the magnitude of the standard deviation. For this problem (where the evaluation was performed using the truth model (T12,10,0.6), in the range of values of q^2 that were examined $(0 \leq q^2 \leq 1)$, the dramatic instabilities observed in Sections 6.2.3 and 6.2.4 do not occur.

Figure (6-27) demonstrates that nearly identical robustness benefits can be gained with the addition of colored noise generated by a first-order shaping filter as with a white noise. However, the addition of colored noise generated by a second-order shaping filter does not have the fast time response of the previous two cases, as shown in Figure (6-28). Nonetheless, the steady-state error and initial overshoot have been substantially improved.

Table (6-6) makes a comparison of the steady-state values of all three aircraft states at the design condition (truth and model dimension equal) for cases with no noise addition, white noise, and first- and

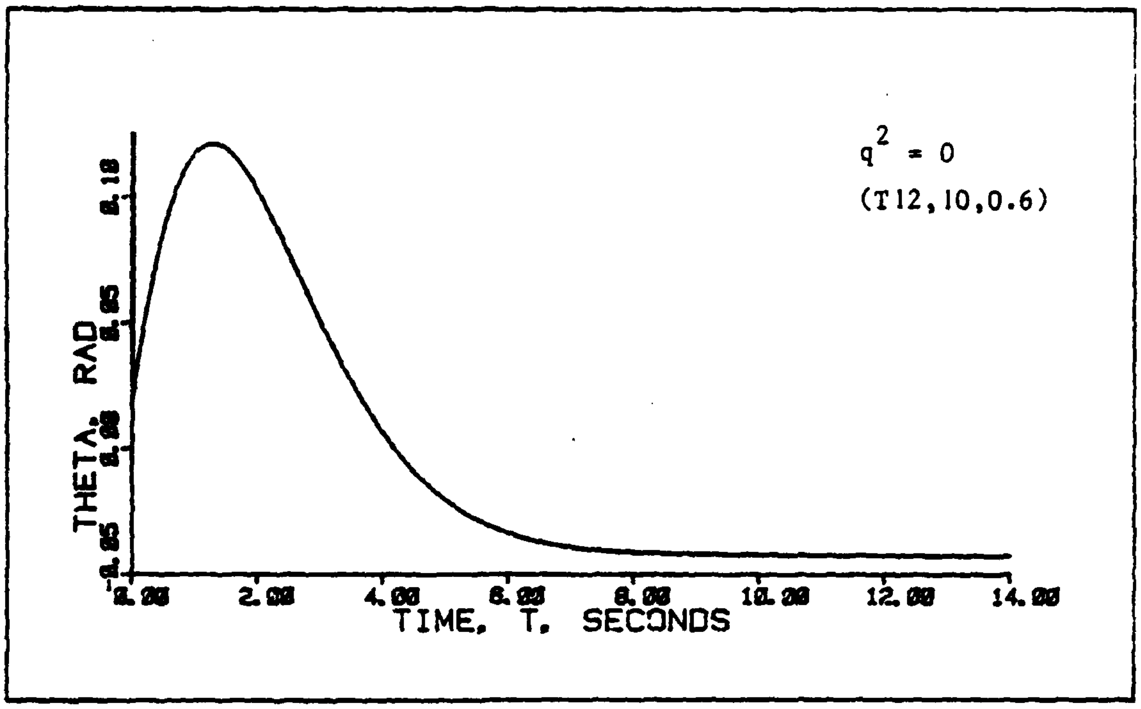


Figure 6-25a: Mean of Θ With No Noise Addition

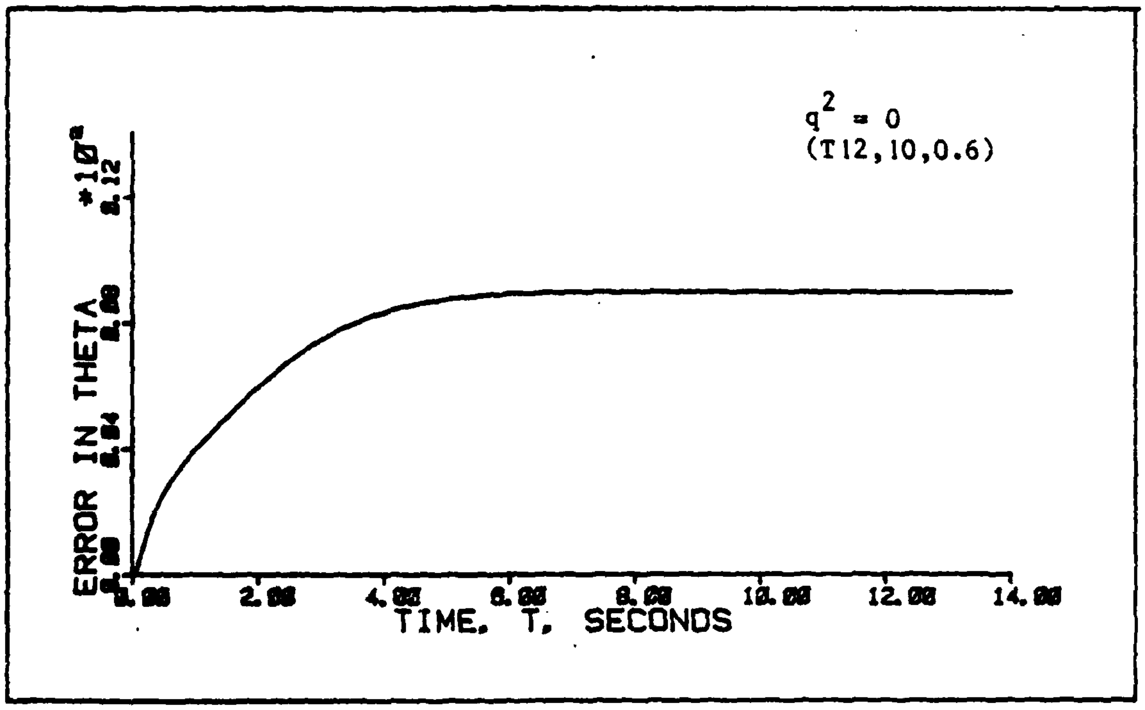


Figure 6-25b: Standard Deviation of Θ With No Noise Addition

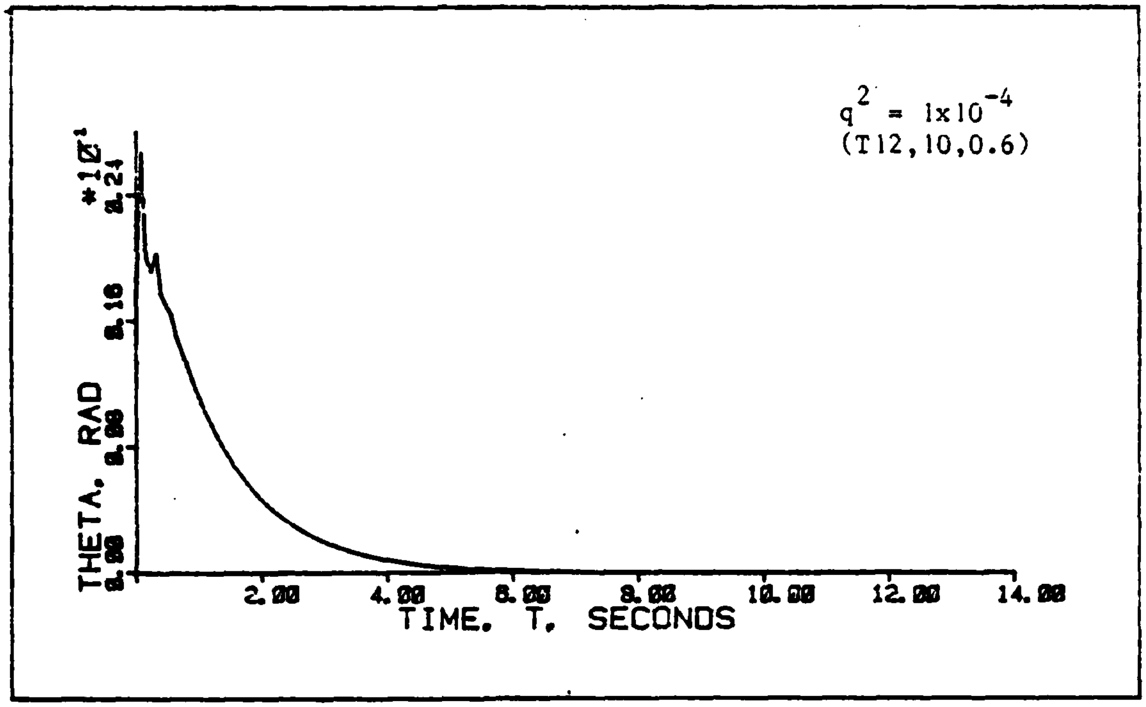


Figure 6-26a: Mean of Θ With White Noise Addition

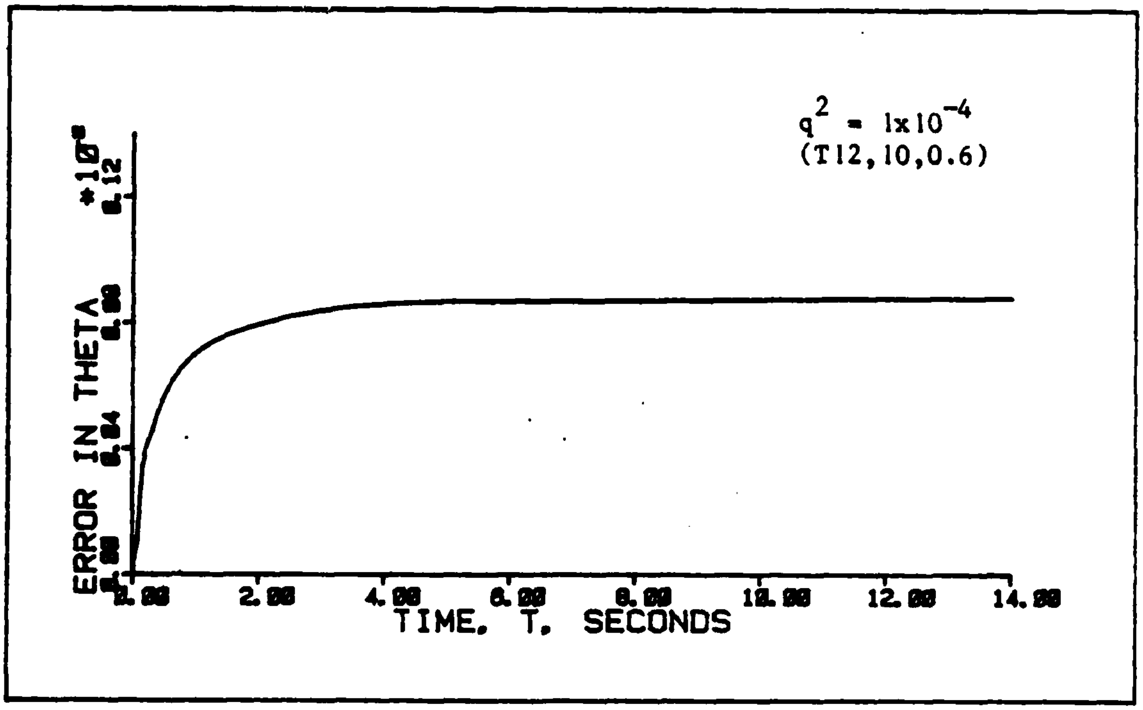


Figure 6-26b: Standard Deviation of Θ With White Noise Addition

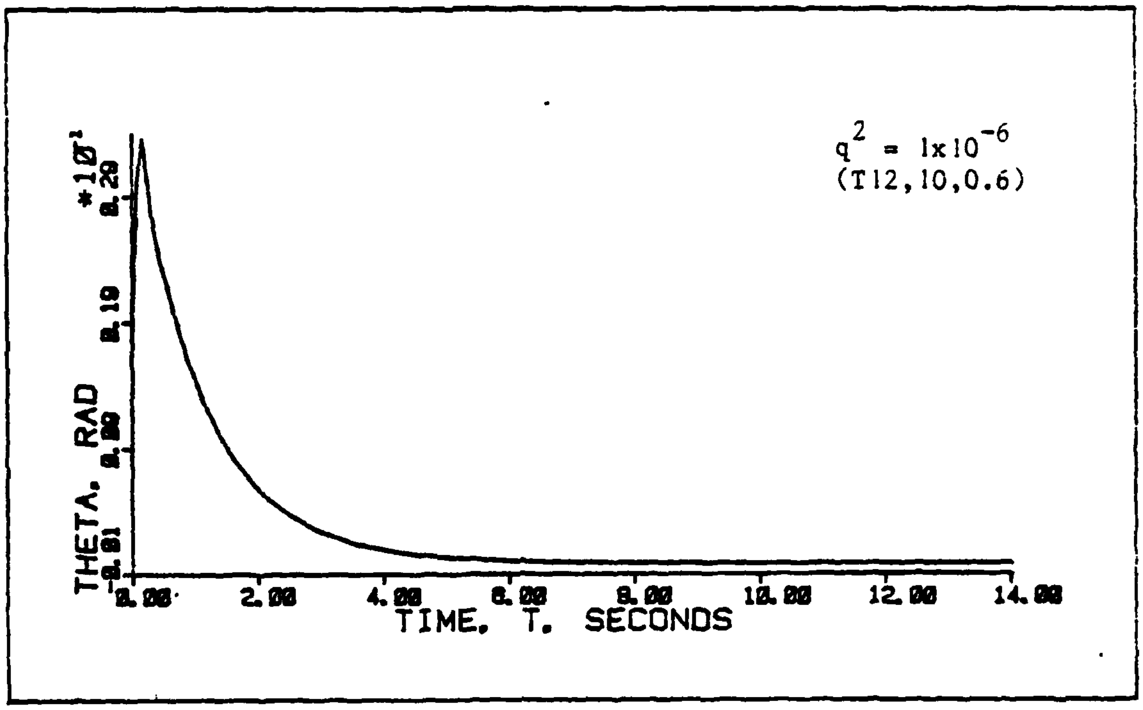


Figure 6-26c: Mean of Θ With White Noise Addition

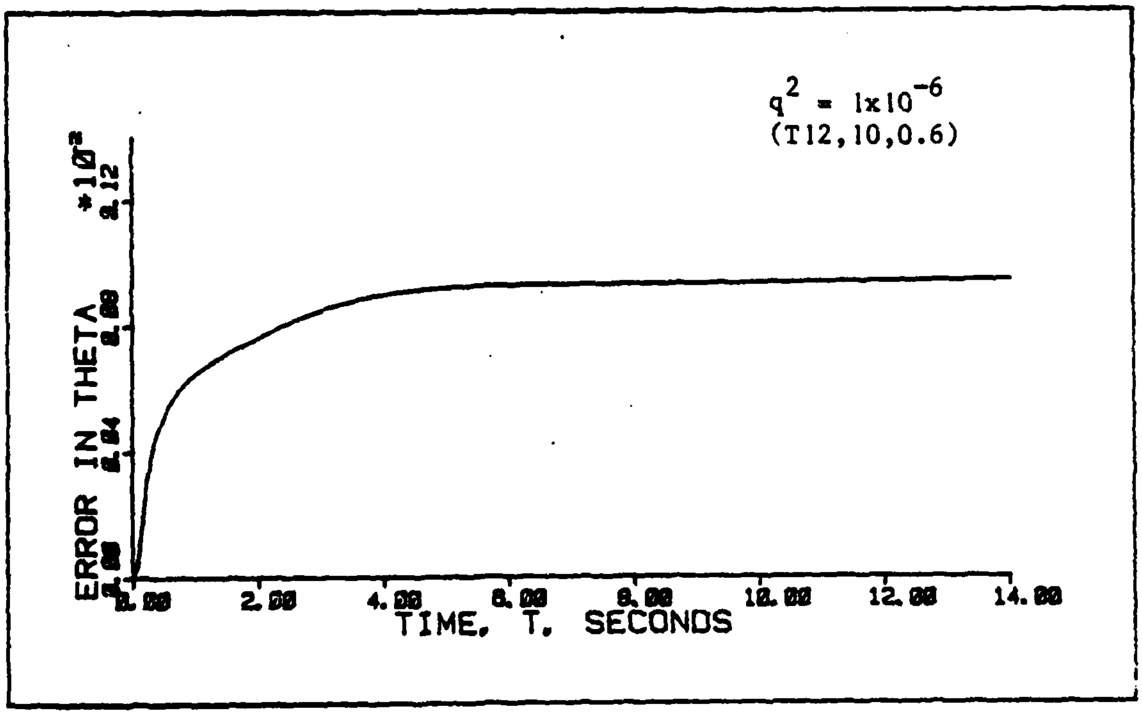


Figure 6-26d: Standard Deviation of Θ With White Noise Addition

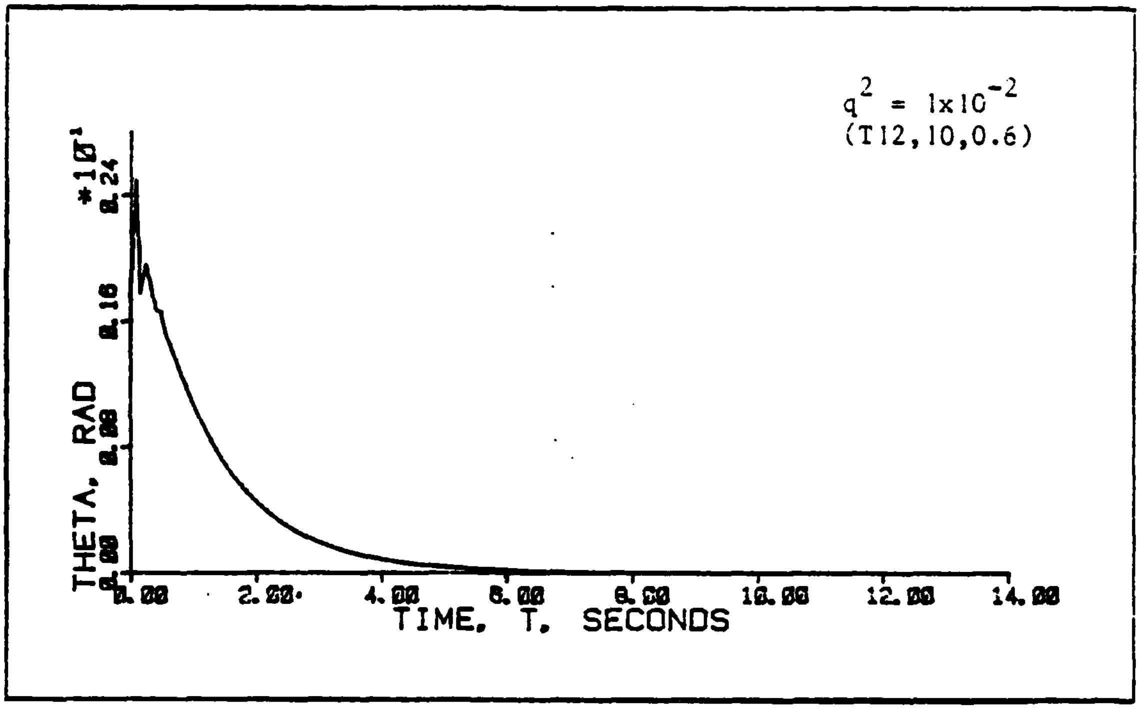


Figure 6-26e: Mean of Θ With White Noise Addition

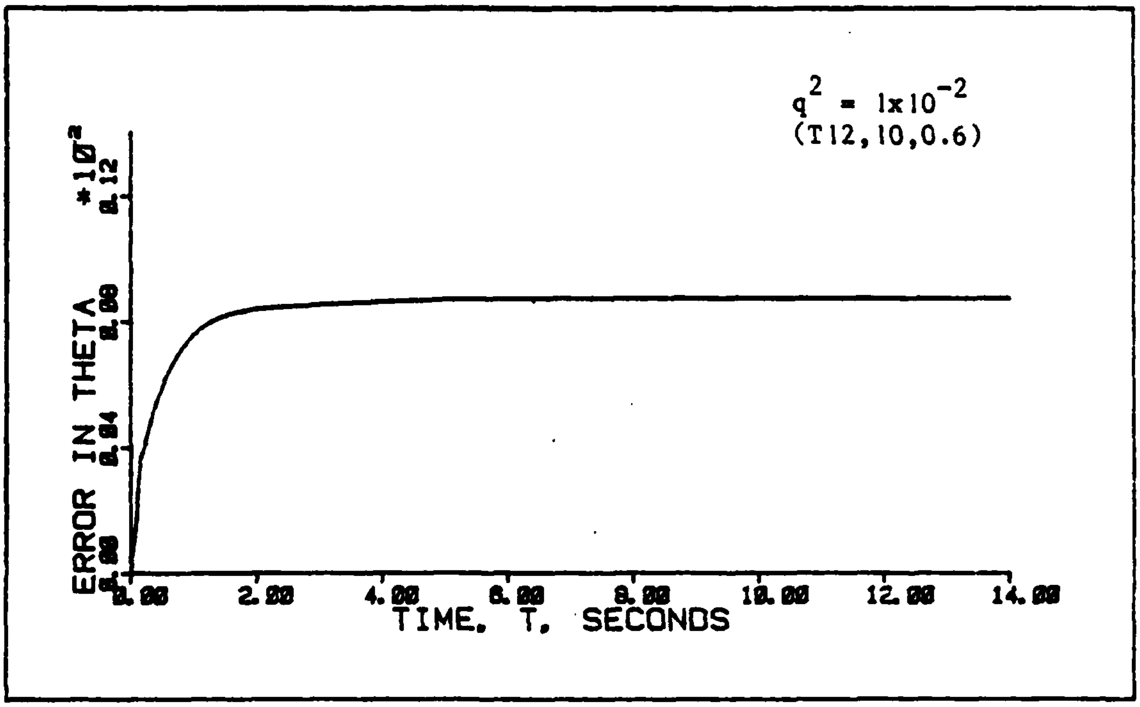


Figure 6-26f: Standard Deviation of Θ With White Noise Addition

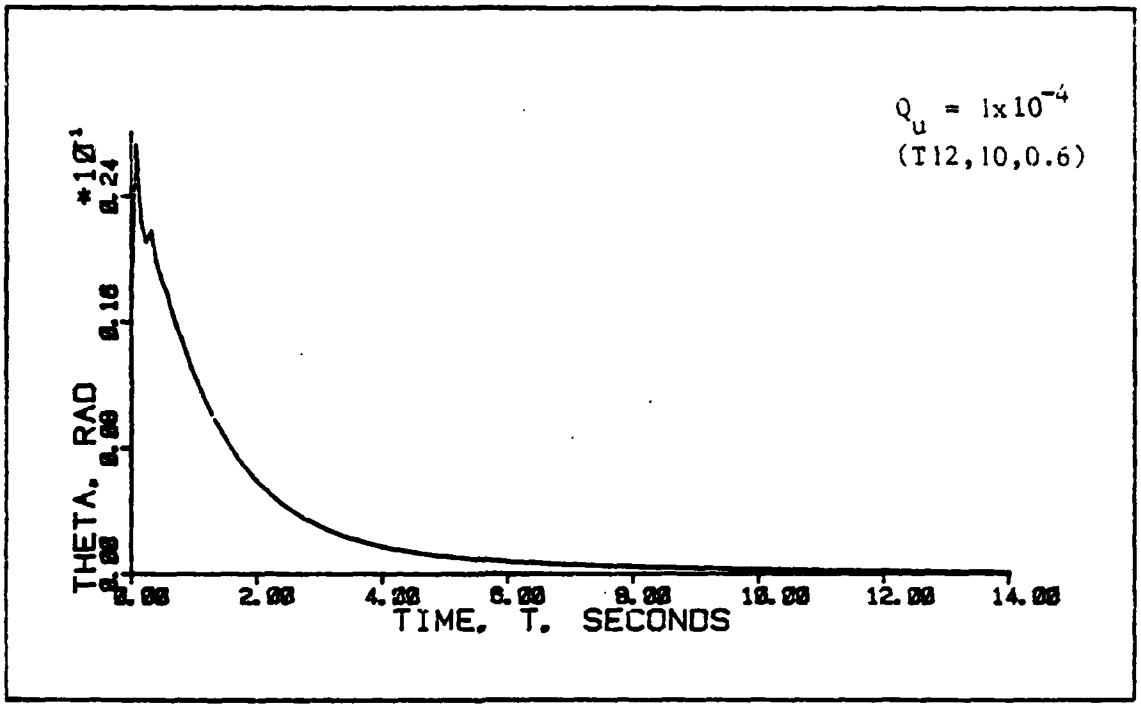


Figure 6-27a: Mean of Θ With First-Order Colored Noise Addition

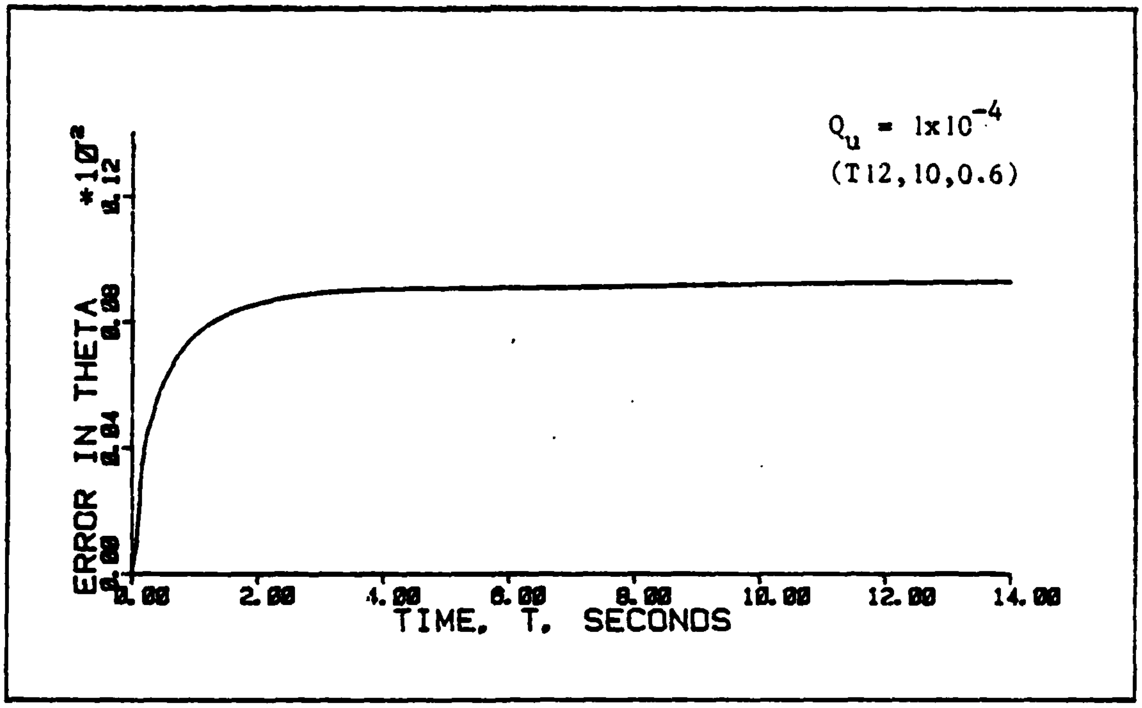


Figure 6-27b: Standard Deviation of Θ With First-Order Colored Noise Addition

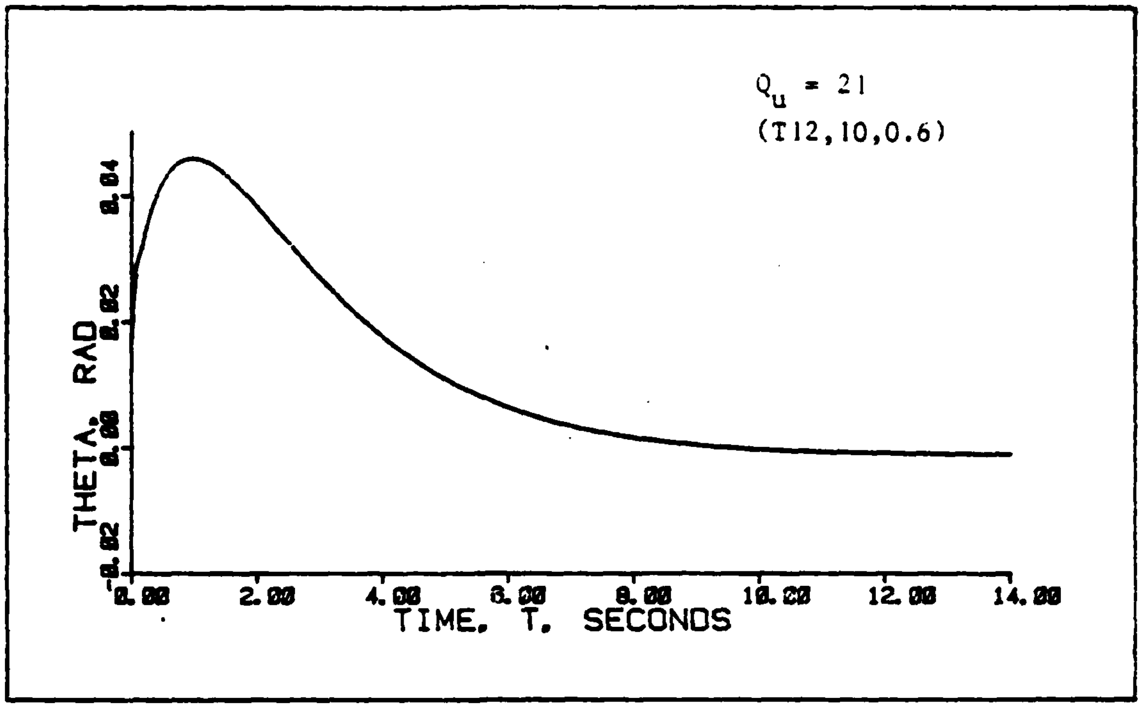


Figure 6-28a: Mean of Θ With Second-Order Colored Noise Addition

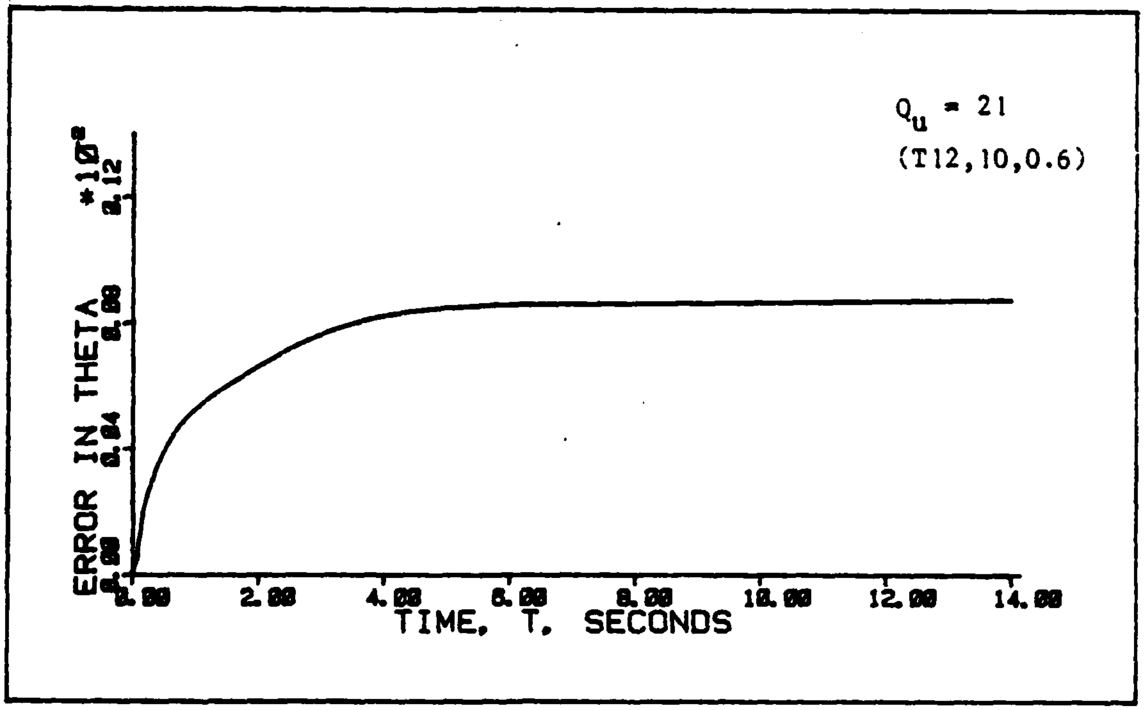


Figure 6-28b: Standard Deviation of Θ With Second-Order Colored Noise Addition

Table 6-6

Comparison of Steady-State Standard Deviations of Aircraft States at the Design Condition For A Sampled-Data System

	q^2	Q_u	σ_Θ	σ_α	σ_q
No Noise	0	-	$8.712\text{x}10^{-4}$	$5.036\text{x}10^{-3}$	$1.206\text{x}10^{-3}$
White Noise	$1\text{x}10^{-4}$	-	$9.583\text{x}10^{-4}$	$5.307\text{x}10^{-3}$	$1.980\text{x}10^{-3}$
1st Order Shaping Filter	-	$1\text{x}10^{-4}$	$1.002\text{x}10^{-3}$	$1.842\text{x}10^{-3}$	$1.765\text{x}10^{-3}$
2nd Order Shaping Filter	-	21	$8.700\text{x}10^{-4}$	$5.040\text{x}10^{-3}$	$1.208\text{x}10^{-3}$

secord-order colored noise. The results are essentially the same as for the continuous-time controller. The same trend is observed in Table (6-7) where higher-order actuator dynamics are included in the truth model. As in the continuous-time case, adding white noise to the model does degrade the performance for all states at the design condition. However, when third-order dynamics are included in the truth model, white input noise decreases the standard deviation Θ. When the filter is evaluated in an environment other than the design condition, adding noise can improve the filter tuning. Decreases in the standard deviations appear in all states for second-order colored noise, and they are not substantial enough to justify the added complexity of the design model. Thus, the Doyle and Stein technique extended to a sampled-data system provides the desired robustification against ignored states for the problem considered in this thesis. Significant performance benefits are not gained by adding time-correlated noise to the system model as opposed to white noise.

Table 6-7

Comparison of Steady-State Standard Deviations
Of Aircraft States with Higher-Order Actuator
Dynamics For A Sampled-Data System

	q^2	Q_u	σ_Θ	σ_α	σ_q
No noise	0	-	$8.984x10^{-4}$	$4.894x10^{-3}$	$1.197x10^{-3}$
White Noise	$1x10^{-4}$	-	$8.831x10^{-4}$	$5.689x10^{-3}$	$4.859x10^{-3}$
1st Order Shaping Filter	-	$1x10^{-4}$	$9.311x10^{-4}$	$5.815x10^{-3}$	$5.143x10^{-3}$
2nd Order Shaping Filter	-	21	$8.789x10^{-4}$	$5.170x10^{-3}$	$2.342x10^{-3}$

6.2.6 Sampled-Data LQG Regulators at Off-Design Condition

As shown in Figure (6-29), the response of the system is unstable at the off-design flight condition with no noise addition.

White noise addition of strength $q^2 = 1x10^{-4}$, as shown in Figure (6-30) is sufficient to recover the stability of the system. First-order colored noise of the same maximum intensity produces a very similar response, as shown in Figure (6-31). However, second-order colored noise of the same maximum intensity is not sufficient to recover stability completely as demonstrated by the divergence of the standard deviation in Figure (6-32). Increasing the maximum noise intensity to $1x10^{-2}$ achieves the desired stability characteristics as shown in Figure (6-33), but the time response is still much slower than for the previous two cases.

It is observed in Figures (6-29) through (6-33) that all three types of input noise substantially improve the robustness properties of the

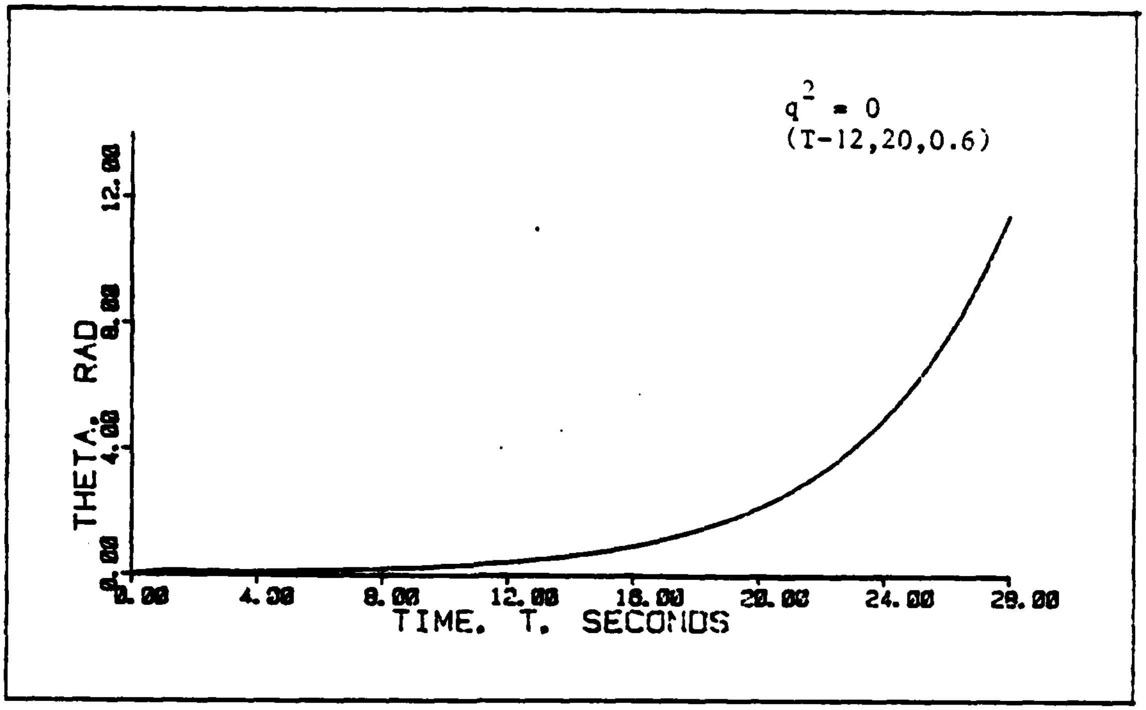


Figure 6-29a: Mean of Θ With No Noise Addition at the Off-Design Flight Condition

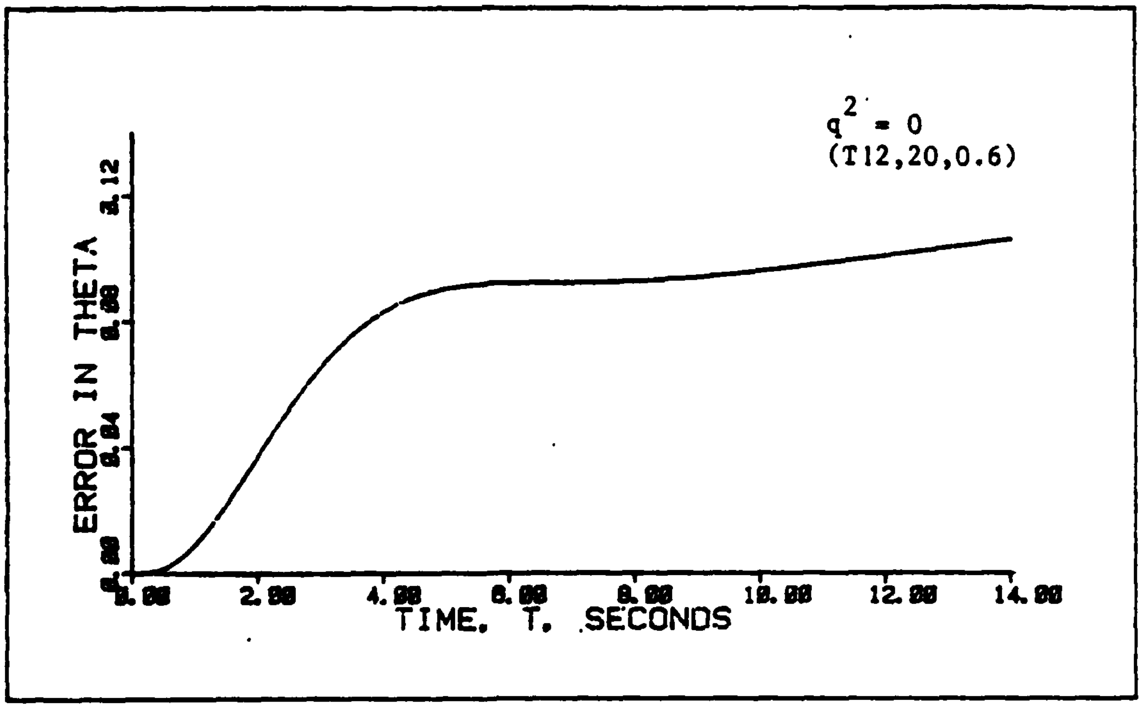


Figure 6-29b: Standard Deviation of Θ With No Noise Addition at the Off-Design COndition

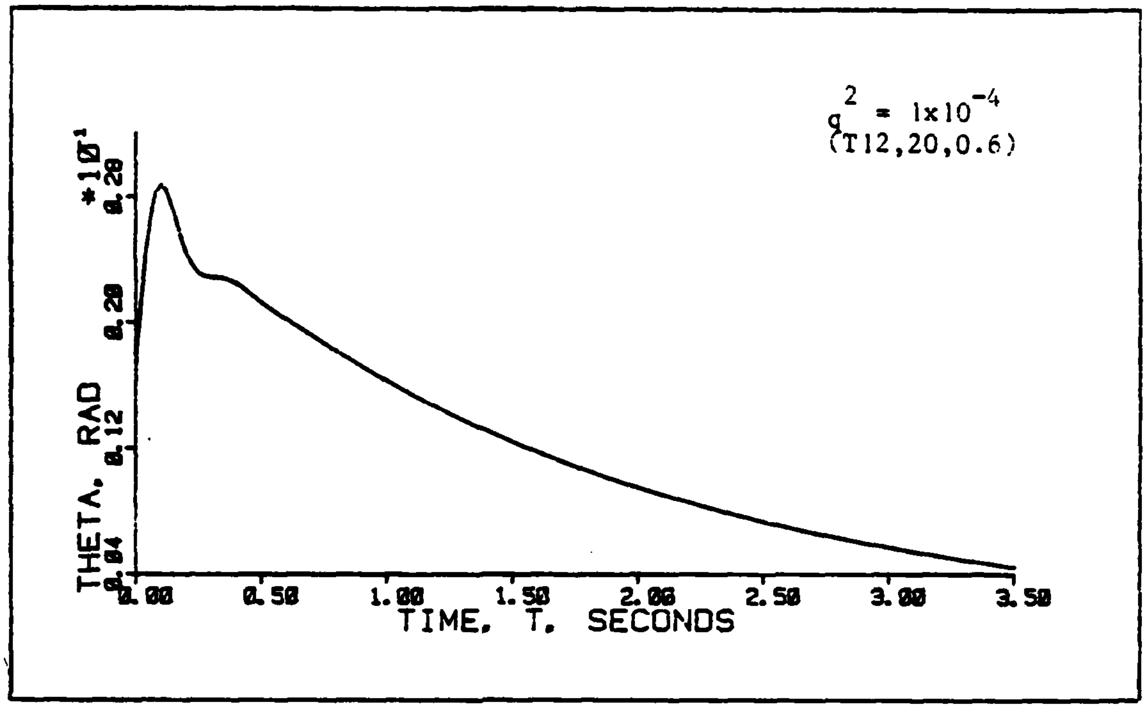


Figure 6-30a: Mean of Θ With White Noise Addition Off-Design Flight Condition

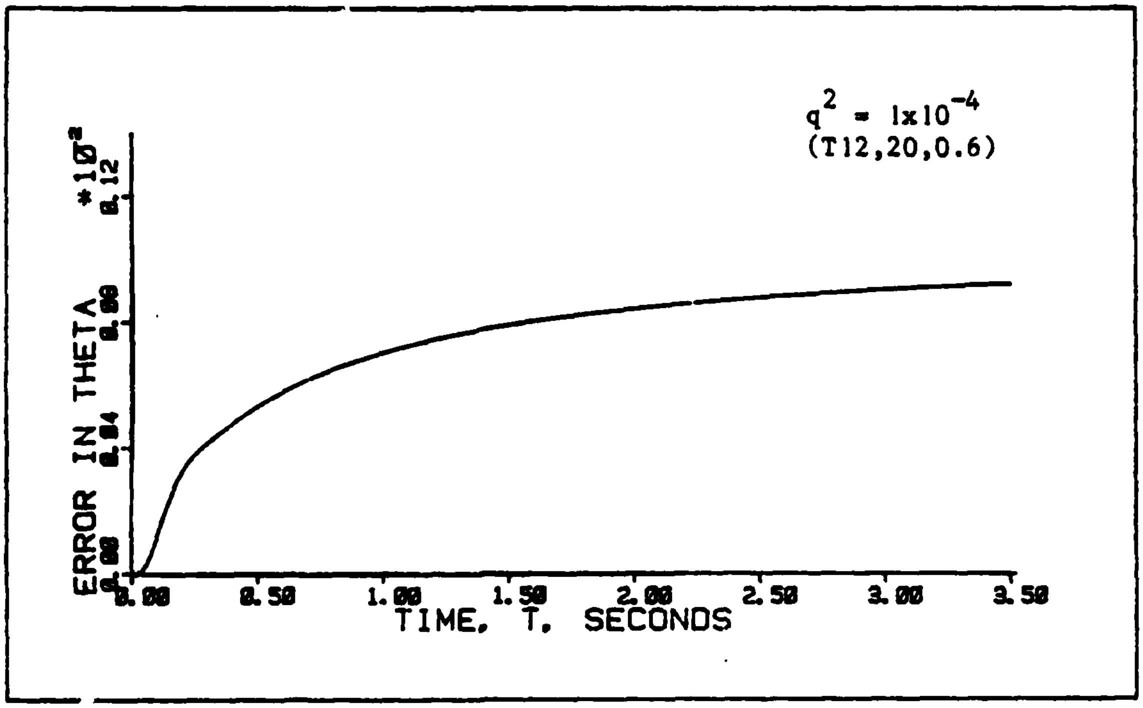


Figure 6-30b: Standard Deviation of Θ With White Noise Addition at the Off-Design Flight Condition

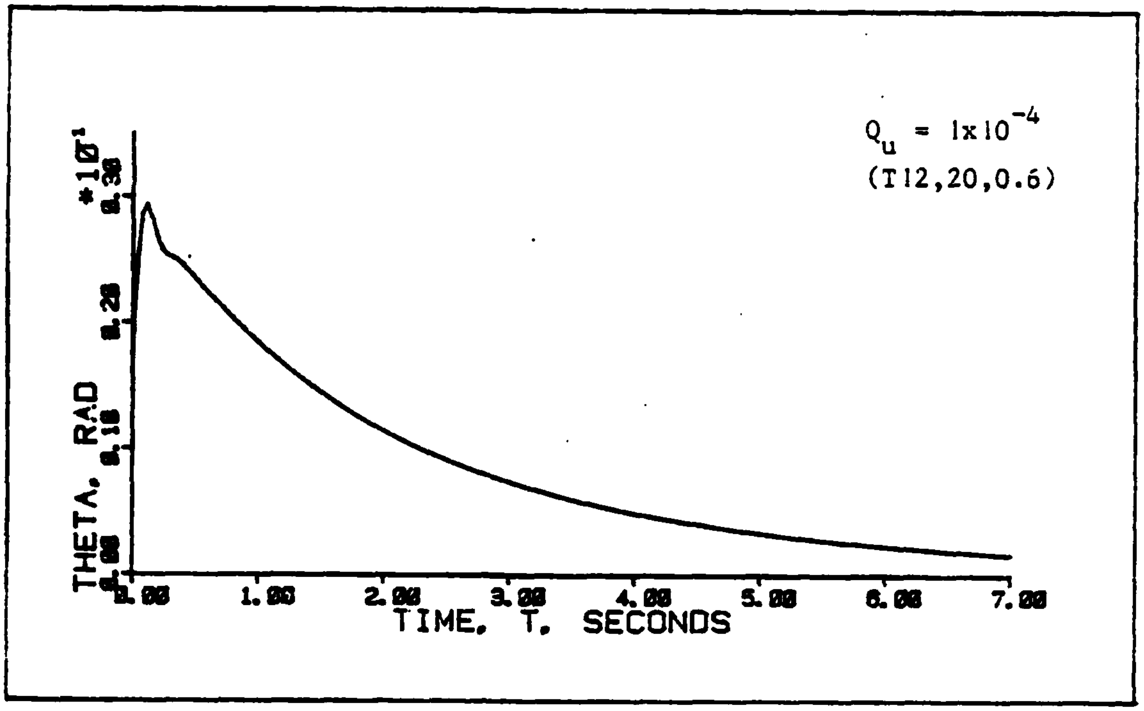


Figure 6-31a: Mean of Θ With First-Order Colored Noise Addition at the Off-Design Flight Condition

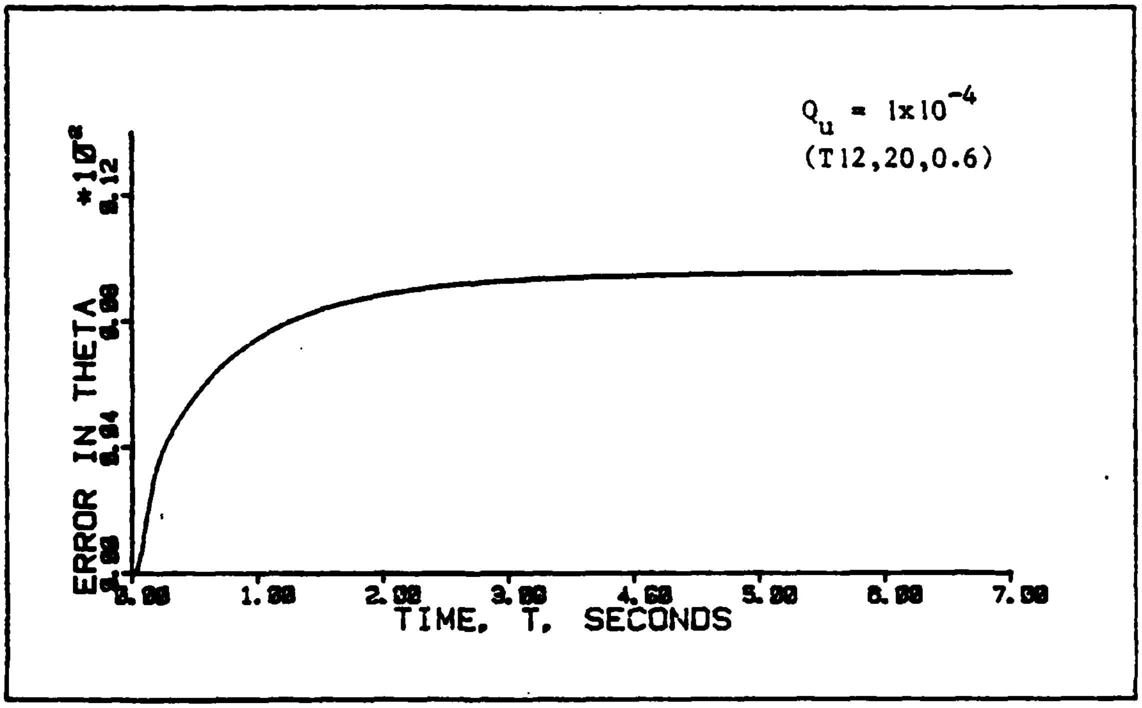


Figure 6-31b: Standard Deviation of Θ With First-Order Colored Noise Addition at the Off-Design Flight Condition

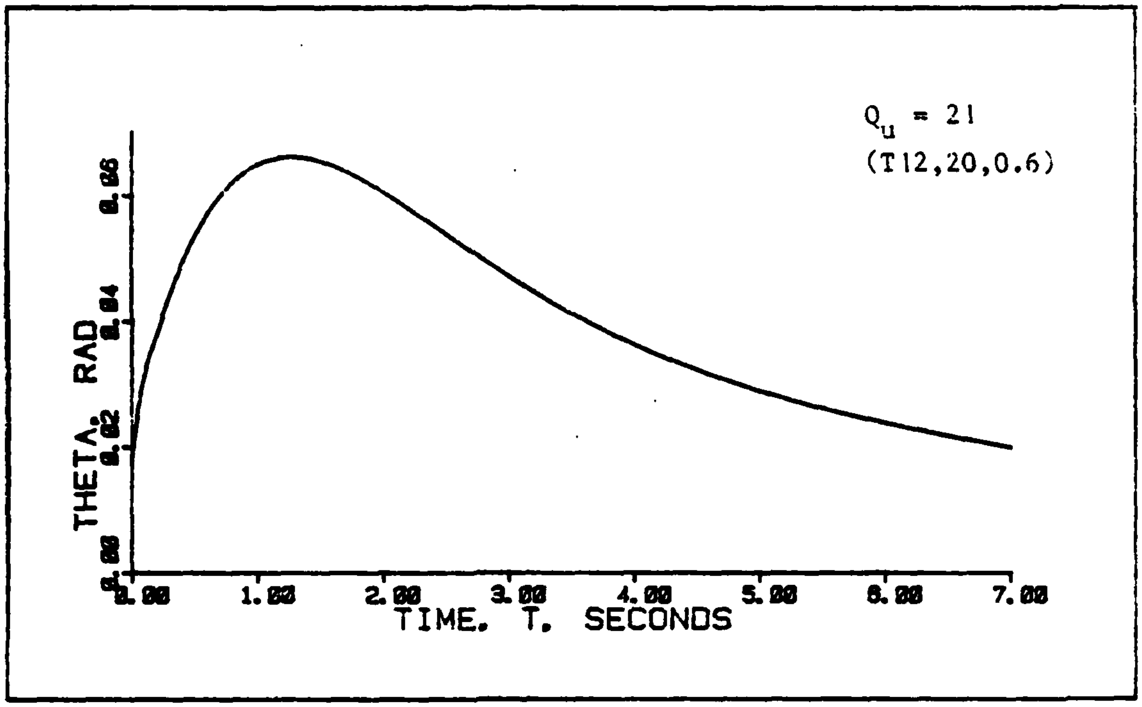


Figure 6-32a: Mean of Θ With Second-Order Colored Noise Addition at the Off-Design Flight Condition

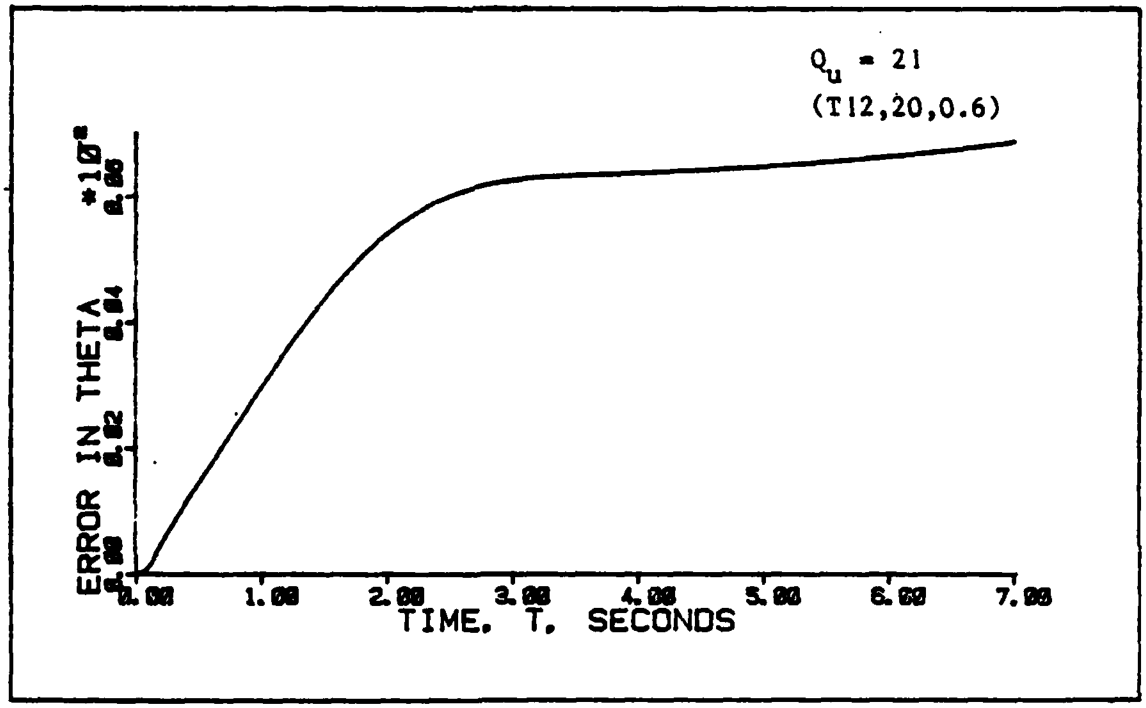


Figure 6-32b: Standard Deviation of Θ With Second-Order Colored Noise Addition at the Off-Design Flight Conditi

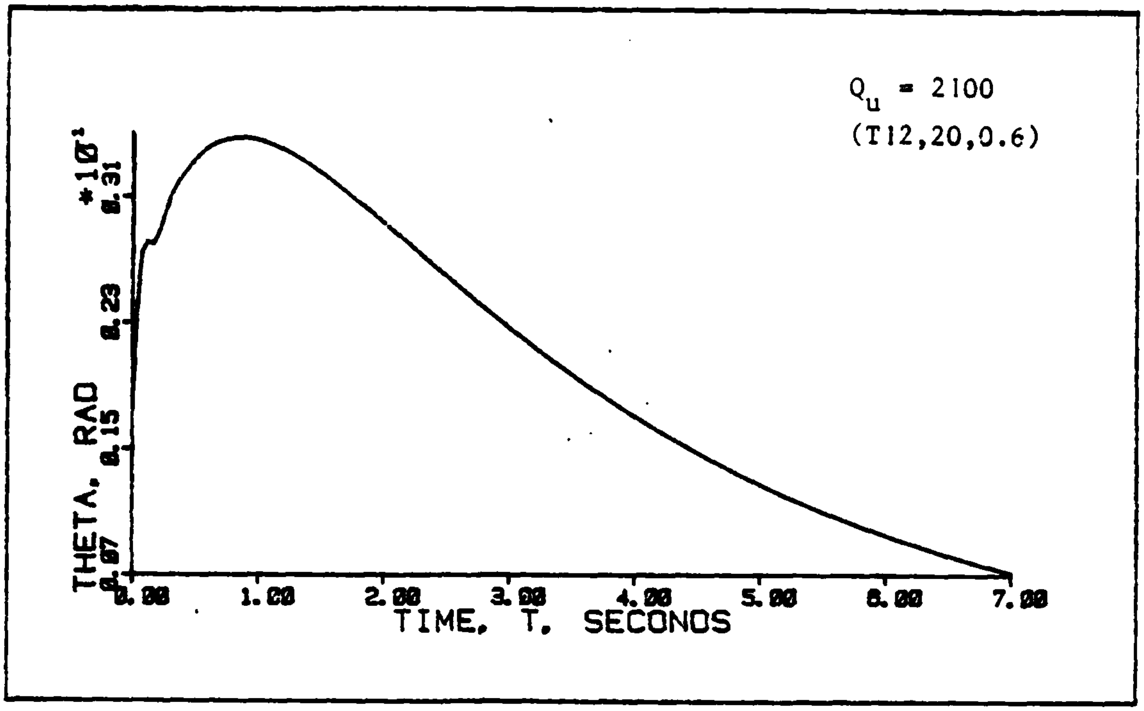


Figure 6-33a: Mean of Θ With Second-Order Colored Noise Addition at the Off-Design Flight Condition

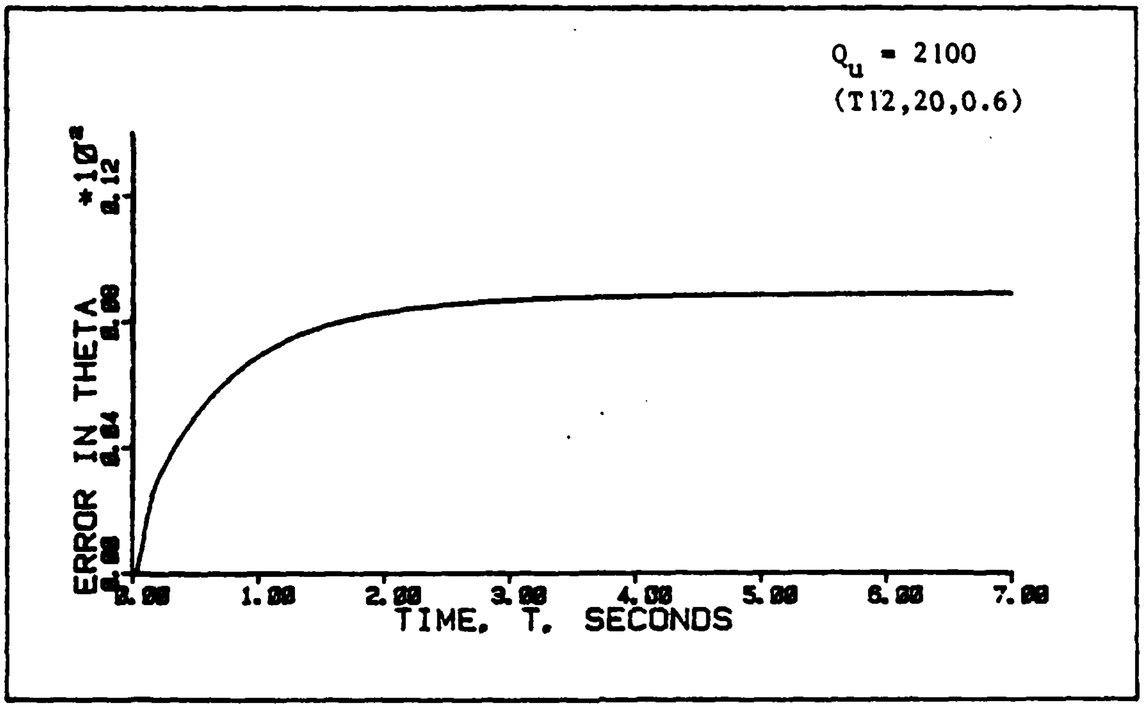


Figure 6-33b: Standard Deviation of Θ With Second-Order Colored Noise Addition at the Off-Design Flight Condition

controlled system when it is subjected to changes in real-world parameters. The improvement due to the addition of white and first-order colored noise is very similar. This indicates that even at the off-design flight condition, the design model is still fairly adequate at low frequencies where the power spectral density of the first-order colored noise is low. Robustness can also be enhanced with second-order colored noise to a lesser degree if the maximum intensity of the noise is sufficiently high. As in the continuous-time case, it is felt that this effect is partially due to the magnitude of the noise at other frequencies, not just the peak near 70 radians/second.

6.3 Sampled-Data PI Controllers

The results of applying the Doyle and Stein technique to a sampled-data PI controller were inconclusive for this problem. However, some results are presented to show how employing a Kalman filter to estimate the states of a system can adversely affect stability.

The software used to design the PI controller allows the design of a full-state feedback controller or one with a Kalman filter in the loop. The performance analysis program then generates time histories of the standard deviation of the states both with and without a filter.

The weighting on the states, controls and control rates are defined in Section 3.2.2 and are given below:

$$X_{11} = \left[\begin{array}{cc|c|cc}
20 & -10 & \text{-0-} & \multicolumn{2}{c}{\text{-0-}} \\
-10 & 10 & & & \\
\hline
\multicolumn{2}{c|}{\text{-0-}} & \text{-0-} & \multicolumn{2}{c}{\text{-0-}} \\
\hline
\multicolumn{2}{c|}{} & & 1 & 0 \\
\multicolumn{2}{c|}{\text{-0-}} & \text{-0-} & & \\
\multicolumn{2}{c|}{} & & 0 & 1
\end{array}\right] \quad (6\text{-}3)$$

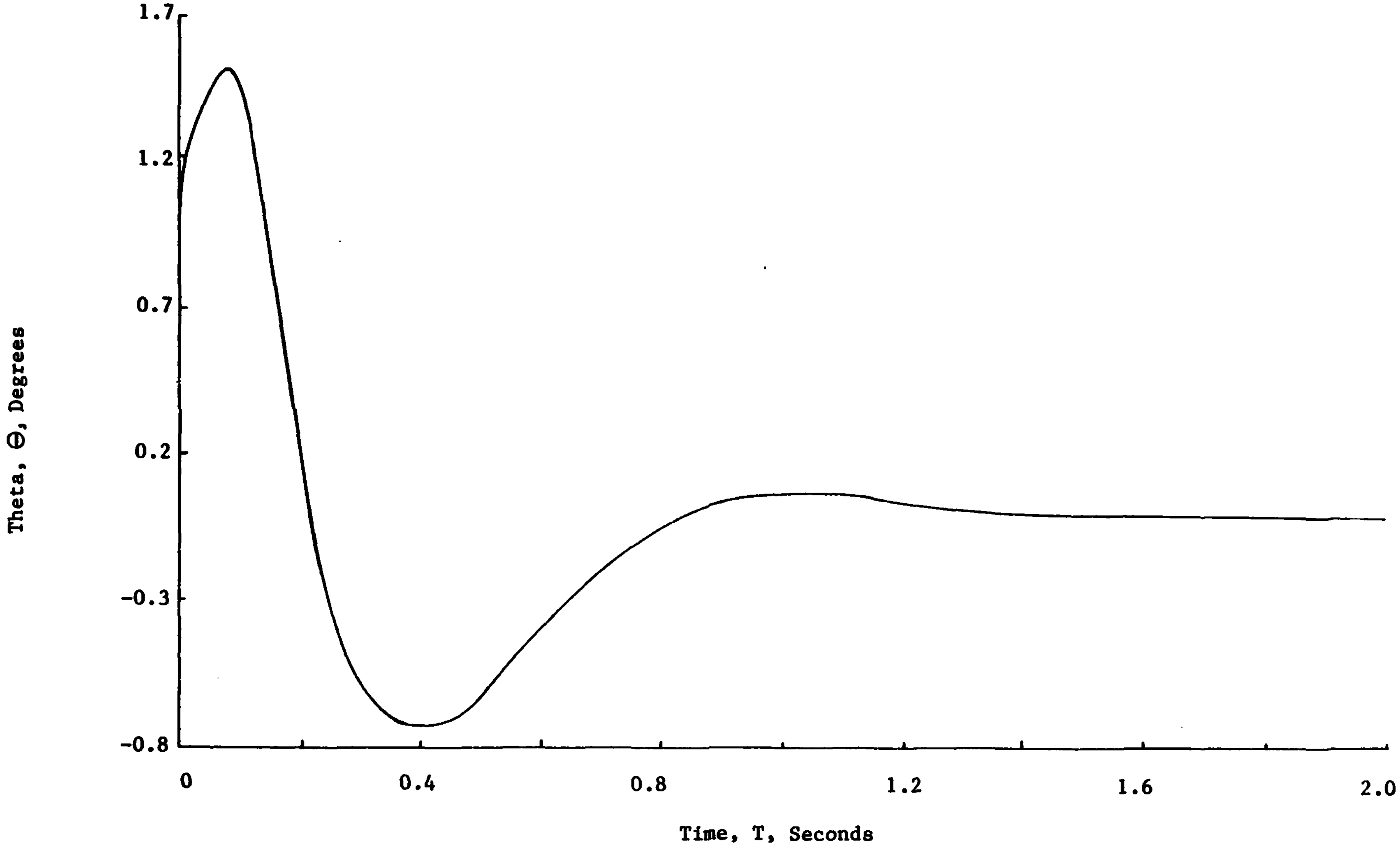


Figure 6-34: Full-State Feedback PI Controller Response of Θ

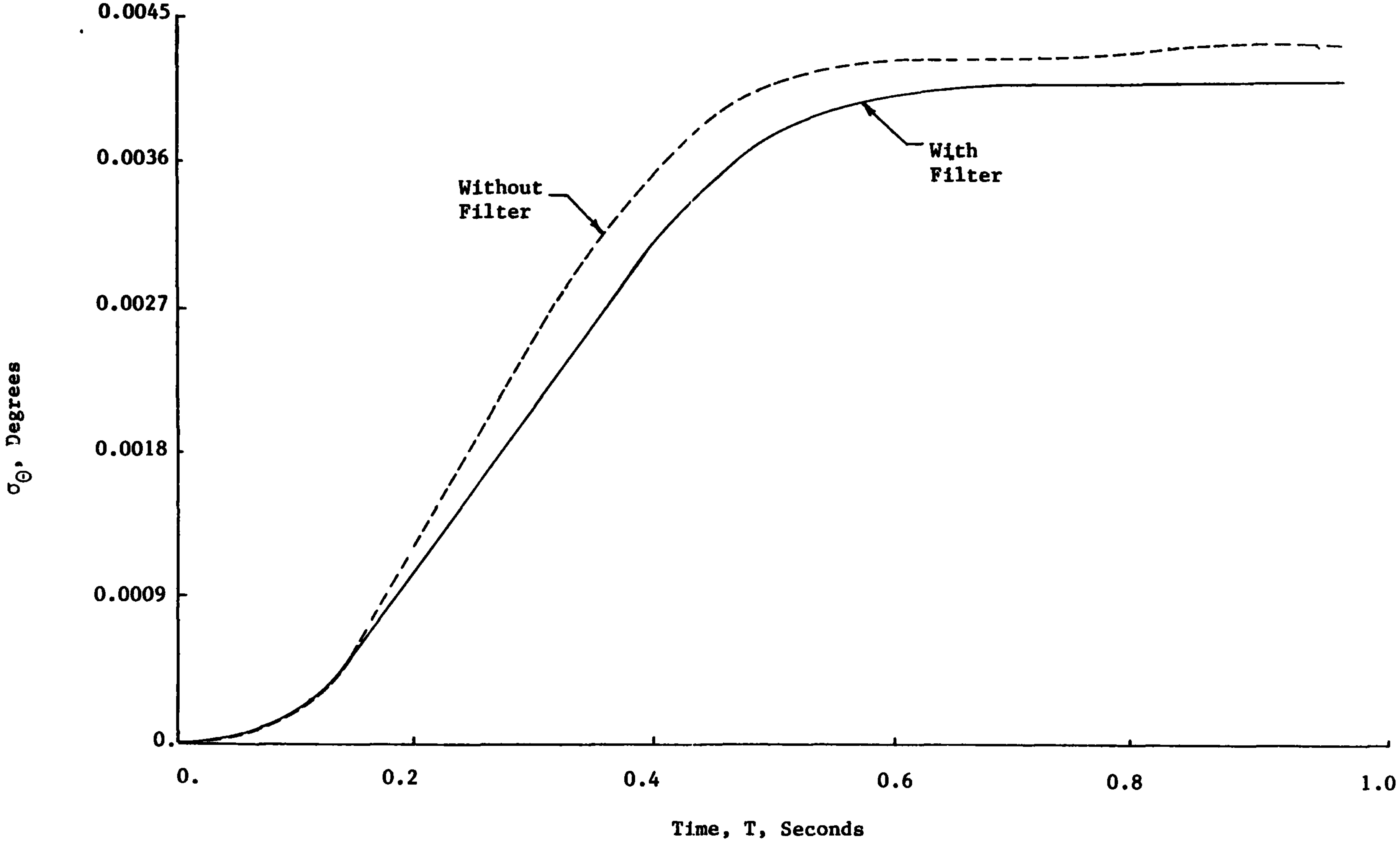


Figure 6-35: Standard Deviation of Θ at Design Condition

$$X_{22} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \qquad (6\text{-}4)$$

$$U = \begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix} \qquad (6\text{-}5)$$

The weighting matrices specified above are given in Reference 27. They are arrived at using a methodology called 'implicit model following". As mentioned before, the software used to design PI regulators also includes a method for designing a Command Generator Tracker to specify feedforward gains that cause the output of the controlled system to track the output of a command model in which the desired characteristics of the response have been built in (damping ratio, overshoot, settling time, etc.). Implicit model following can be used to affect the feedback gains of the PI controller in a way that makes the controlled system more robust. A more detailed explanation of implicit model following is given in References 27 and 29.

Figure (6-34) shows the time-response of the full-state feedback system with an initial condition of one degree on Θ evaluated against a deterministic truth model of the same dimension. It is seen that the response has very nice, well-damped second-order characteristics. Figure (6-35) demonstrates the standard deviations for this controller in a stochastic environment are very similar with and without a filter in the loop. The design model is defined in Section 5.4. In Figure (6-36),

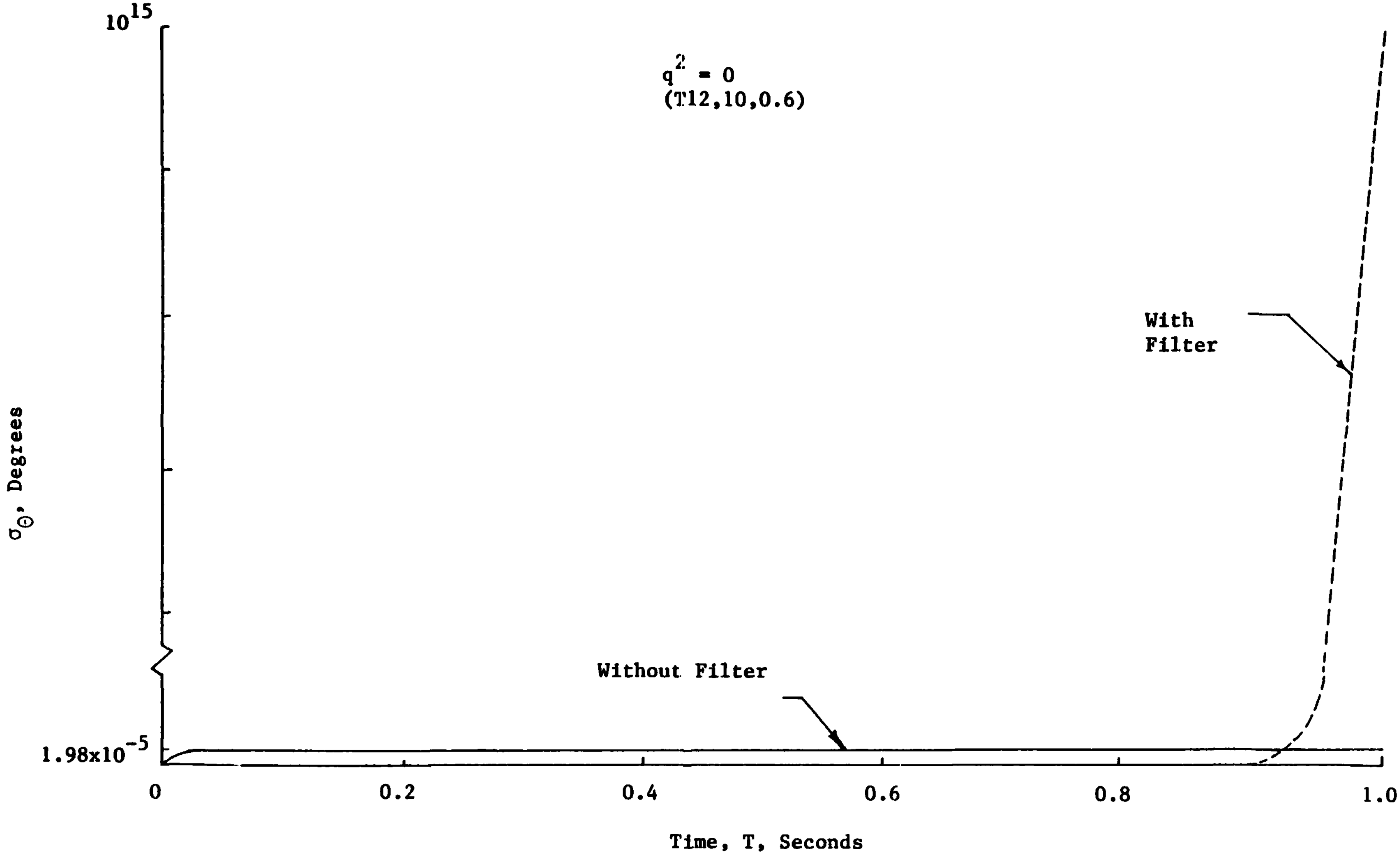


Figure 6-36: Standard Deviation of Θ Evaluated Against Third-Order Actuator Dynamics

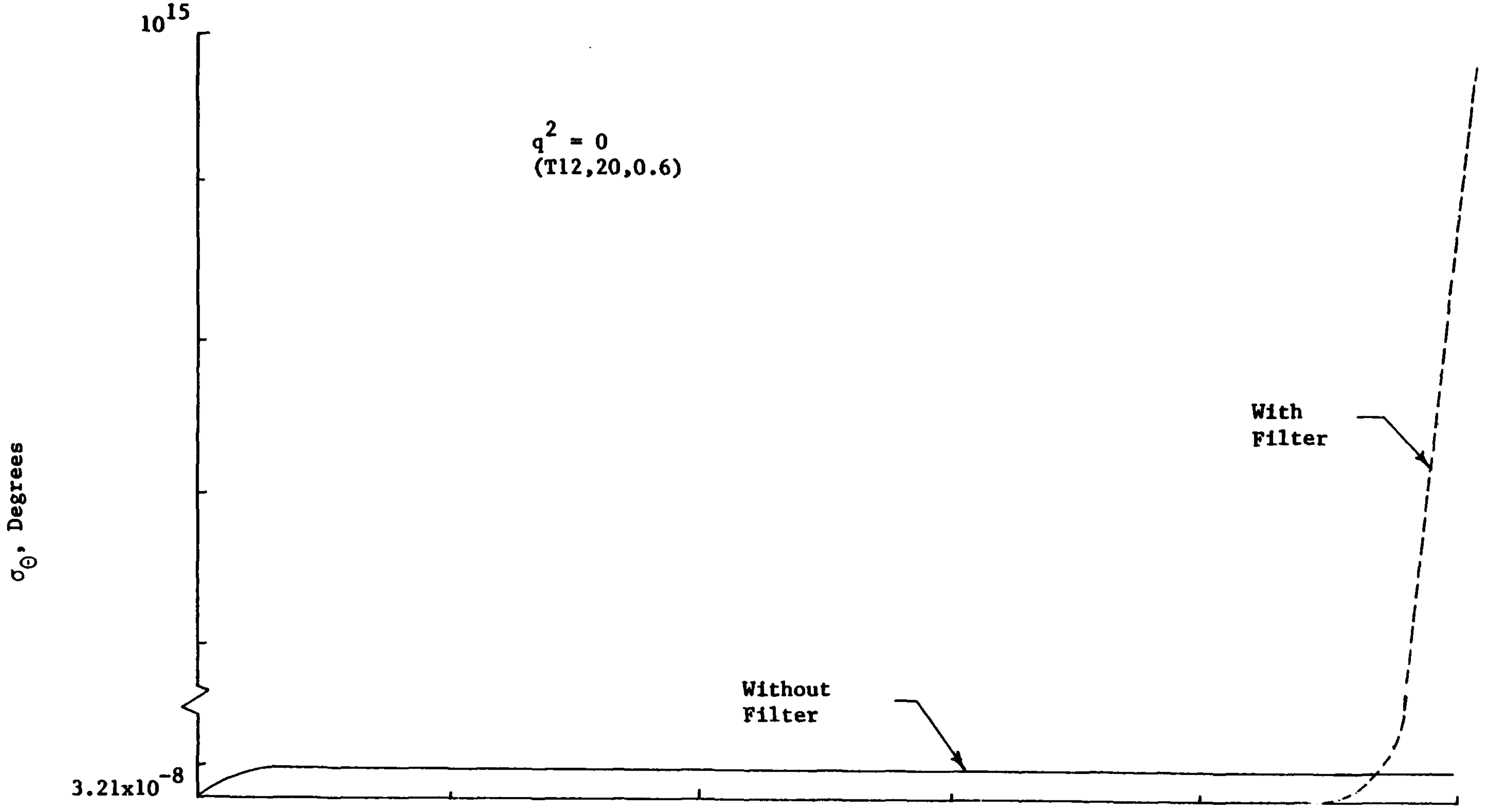


Figure 6-37: Standard Deviation of Θ At an Off-Design Flight Condition

the results of the performance analysis are shown, where the eight-state controller was evaluated against a truth model with higher order actuator dynamics in the truth model. It is seen that without full-state feedback, stability was lost. The results are the same at an off-design condition (T12,20,0.6) as shown in Figure (6-37). Thus, the figures demonstrate that when actuator states are ignored in the design model, a full-state feedback controller exhibits good stability robustness properties. However, once a filter is inserted into the loop, these properties are lost.

Attempts were made to recover the stability of the controlled system by applying the Doyle and Stein technique. It was found that any non-zero value of q^2 only made the standard deviations diverge more rapidly. This was true for both flight conditions.

Recall that the Doyle and Stein technique was applied to sampled-data LQG regulators by modifying the Q_u matrix as follows:

$$Q_d(q) = Q_{do} + q^2 BVB^T \Delta t \tag{6-6}$$

using the B matrix of the continuous-time system equation.

The technique was applied to sampled-data PI controllers with the equation

$$Q_d(q) = Q_{do} + q^2 B_d V B_d{}^T / \Delta t \tag{6-7}$$

using the B_d matrix of the equivalent discrete-time system equation.

The Doyle and Stein technique was derived by equating the return-difference mappings for a full-state feedback system and a filter-based system. The derivation was based on the controllers being a full-state feedback regulator and an LQG regulator. It was expected, since the Kalman filter is the same for LQG regulators and PI controllers, that at least some robustness enhancement could be obtained by directly applying the technique to a PI controller. However, this was not found to be true.

6.4 Summary

This chapter has presented the results of applying two techniques to improve the robustness properties of a controlled system. The two methods examined are the techniques of inputting stationary white or time-correlated Gaussian noise into a system model during the process of tuning the Kalman filter.

Two separate issues of robustness were examined. The first was the idea of robustifying a controller so that instabilities do not occur when states are ignored inthe controller and Kalman filter design model. For this thesis, the problem considered was an aircraft flight control problem, and the effect of ignoring states that described the actuator dynamics was examined. In this particular instance, the design model misrepresentation was confined to a particular portion of the frequency spectrum; this specifically motivated investigation of adding time-correlated rather than white noise for robustification.

The second robustness issue was the idea of robustifying a controller against changes in the real-world parameters upon which the controller design was based. Here the designed model misrepresentation was not confined to only a particular band of frequencies.

The primary result of this chapter is that the stability robustness of the controlled system can be substantially enhanced by applying both robustification techniques. However, for the problem considered, the Doyle and Stein technique of inputting white noise into the system model is the most appropriate way to achieve the robustness enhancement.

Conclusions about the results presented in this chapter are made in the following chapter. In addition, some recommendations for further research are made.

VII. Conclusions and Recommendations

The robustification of LQG regulators by inputting white noise or time-correlated noise generated by a first-order shaping filter into the system model did work well for all three applications of the controllers examined (continuous, discretized, sampled-data). In addition, time-correlated noise generated by a second-order shaping filter improved the controller's robustness properties to a lesser degree for the continuous-time and sampled-data case. For the discretized case, this method produced instabilities in the response of the system and could not be used.

It was noted that in Chapter VI that few performance benefits were gained by using time-correlated noise as opposed to white noise for the problem considered in this thesis. It is felt that the range of frequencies where the design model differed from the chosen representation of the real-world was not sufficiently narrow in this case to result in performance benefits. Greater performance benefits may be realized by examining the application of colored noise addition to problems such as gust-load alleviation, flutter suppression, or aeroelastic effects such as including first- and higher-order bending modes in the truth model representation of the real world.

It is felt that the robustification techniques for LQG regulators were examined in great detail in this thesis. However, the extension to PI controllers was only touched upon, and the Doyle and Stein technique was applied unsuccessfully. To examine this more thoroughly, it is suggested that the Doyle and Stein technique be applied more directly to PI controllers. That is, the conditions analogous to those of Doyle and Stein that would make the return-difference mapping asymptotically equal for a full-state feedback controller and a filter-based PI controller should be derived.

The technique of injecting time-correlated noise into the system model for a PI controller was not examined in this thesis. The existing software was not originally designed to allow augmenting of states with the design model. Therefore, it would be difficult to augment shaping filter states to allow for colored input noise. The modifications shown in Appendix C were made so that a few cases of colored input noise could be examined quickly. If an extensive study were made, the awkwardness of these modifications would become apparent. To examine all of the robustification techniques discussed in this thesis would thus require additional software.

It has been mentioned that a drawback of adding colored-noise to a system model is that it adds states to the Kalman filter model. An alternative method, called residualization (Ref 28), reduces the order of the model back to that of the original system before the shaping filter states were added. It would be useful to examine this implementation and compare it to the performance of the implementation used in this thesis for a problem where colored input noise is appropriate.

Finally, as noted in Sections 6.2.3 and 6.2.4, the technique of adding white or colored noise to a continuous-time controller, then discretizing the controller only improved robustness characteristics for a finite range of q^2 or Q_u. Beyond that range, the closed-loop system was unstable. This phenomenon is not well understood, and further research would be warranted to determine why this occurs. While it is not expected that an extension of the Doyle and Stein method would have all the same characteristics of the original technique, it would be useful to determine what occurs during the discretization process to drive closed-loop system poles outside the z-domain unit circle so abruptly.

The results of the previous chapter have thus demonstrated that the techniques considered can substantially improve the robustness properties of a controlled system. For the particular problem considered white noise added to the system model at the control entry points is the appropriate method for accomplishing the robustification

BIBLIOGRAPHY

1. Anderson, B.D. and J.B. Moore. Linear Optimal Control. Englewood Cliffs, New Jersey: Prentice-Hall, 1971.

2. Barfield, A.F. Aircraft Control Engineer. Unpublished AFTI/F-16 Linear Aerodynamic Data. Air Force Flight Dynamics Laboratory, Wright-Patterson Air Force Base, Ohio, October 1980.

3. Chalk, C.R. et al. "Background Information and User Guide for MIL-F-8785B (ASG) Entitled Military Specifications--Flying Quality of Piloted Airplanes", AFFDL-TR-69-72, Air Force Flight Dynamics Laboratory, Wright-Patterson Air Force Base, Ohio, August, 1969.

4. D'Azzo, J.J. and C.H. Houpis. Linear Control System Analysis and Design. New York: McGraw-Hill Book Company, 1975.

5. Doyle, J.C. "Guaranteed Margins for LQG Regulators", IEEE Transactions on Automatic Control, AC-23 (4). 756-757, August, 1978.

6. Doyle, J.C. and G. Stein. "Robustness with Observers", IEEE Transactions on Automatic Control, AC-24 (4). 607-611, August, 1979.

7. Durrett, J.C. "A Beginning Primer on $\dot{x} = Ax + Bu$ for the Airplane", AFFDL-FGC-TM-72-13, Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio, November, 1972.

8. Durrett, J.C. "On $\dot{x} = Ax + Bu + m\eta$ for the Airplane, Wind Gust and Pilot", AFFDL/FGC-TM-72-21, Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio, November 1972.

9. Durrett, J.C. "On $\dot{x} = Ax + Bu$ for the Airplane: Simplification of the General Linear Equations of Motion", AFFDL/FGC-TM-72-24, Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio, November 1972.

10. Etkin, B. Dynamics of Atmospheric Flight. New York, New York: John Wiley and Sons, Inc., 1972.

11. Floyd, R.M. Aircraft Control Engineer. Unpublished AFTI/F-16 Aerodynamic Stability Derivative Data. Air Force Flight Dynamics Laboratory, Wright-Patterson Air Force Base, Ohio, June, 1980.

12. Floyd, R.M. "Design of Advanced Digital Flight Control Systems via Command Generator Tracker (CGT) Systhesis Methods, Volume 1". M.S. Thesis, Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio, December, 1981.

13. Floyd, R.M. "Design of Advanced Digital Flight Control Systems via Command Generator Tracker (CGT) Synthesis Methods, Volume 2". M.S. Thesis, Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio, December, 1981.

14. Floyd, R.M. Doctoral Student (personal interviews). Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio, July 1983 - December 1983.

15. Gangsaas, D. and U. Ly. "Application of a Modified Linear Quadratic Gaussian Design to Active Control of a Transport Airplane", AIAA Paper No. 79-1746, August, 1979.

16. Gangsaas, D., U. Ly, and D.C. Norman. "Practical Gust Load Alleviation and Flutter Suppression Control Laws Based on an LQG Methodology", AIAA 195h Aerospace Sciences Meeting. St. Louis, Missouri: Paper 81-0022, January 1981.

17. Heath, R.E. "State Variable Model of Wind Gusts", AFFDL/FGC-TM-72-12, Air Force Flight Dynamics Laboratory, Wright-Patterson Air Force Base, Ohio, July, 1972.

18. Kleinman, D.L. "A Description of Compuer Programs Useful in Linear System Studies", Tec Rep. TR-75-4, University of Connecticut, Storrs, Connecticut, October, 1975.

19. Kwakernaak, H. and R. Sivan. Linear Optimal Control Systems. New York, New York: John Wiley and Sons, Inc., 1972.

20. Larimer, S.J. "An Interactive Computer-Aided Design Program for Digital and Continuous System Analysis and Synthesis", (TOTAL), M.S. Thesis, Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio, December, 1978.

21. Lloyd, E.D. "Robust Control Systems", M.S. Thesis, Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio, December, 1981.

22. Maybeck, P.S. Professor of Electrical Engineering (personal interviews). Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio, April, 1983 - December 1983.

23. Maybeck, P.S. Stochastic Models, Estimation, and Control, Volume 1. New York, San Francisco and London: Academic Press, 1979.

24. Maybeck, P.S. Stochastic Models, Estimation, and Control, Volume 3. New York, San Francisco and London: Academic Press, 1982.

25. McRuer, D., I. Ashkenas, and D. Graham. Aircraft Dynamics and Automatic Control. Princeton, New Jersey: Princeton University Press, 1973.

26. McMillian J.P. "Command Generator Tracker Synthesis Methods Using an LQG-Derived Proportional-Plus-Integral Controller Based on the Integral of the Regulation Error", M.S. Thesis, Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio, December, 1983.

27. Miller, W.G. "Robust Multivariable Controller Design via Implicit Model Following Methods", M.S. Thesis, Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio, December 1983.

28. Moore, J.B., D. Gangsaas, and J.D. Blight. "Performance and Robustness Trades in LQG Regulator Design", Proceedings of IEEE Conference on Decision and Control. 1191-1200, 1981.

29. Mosely, A. "Design of Advanced Digital Flight Control Systems via Command Generator Tracker (CGT) Synthesis Methods, Volume 1", M.S. Thesis, Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio, December, 1982.

30. Mosely, A. "Design of Advanced Digital Flight control Systems via Command Generator Tracker (CGT) Synthesis Methods, Volume 2", M.S. Thesis, Air Force Institute of Technology, Wright-Patterson Air force Base, Ohio, December, 1982.

31. Roskam, J. Airplane Flight Dynamics and Automatic Flight Controls. Lawrence, Kansas: Roskam Aviation and Engineering, 1979.

APPENDIX A: Generic Controller Format

A1. Introduction

In Chapter II, the idea of a standard format for a linear controller was introduced. The format is particularly useful in that any LQG controller synthesized by LQG methods can be rearranged into the standard form. Subsequently, the performance analysis equations presented in Chapter II apply to any controller put in the "generic" form, not just LQG regulators.

This appendix defines explicitly the generic form controller equations for the types of controllers considered in this thesis. In the first section, the continuous-time LQ regulator is examined, that is, a full-state feedback regulator with a control law expressed as:

$$\underline{u}^*(t) = G_c^* \, \underline{x}(t) \tag{A-1}$$

The full-state feedback generic structure is presented because it is desired to recover the robustness characteristics of the LQ controller with the addition of input noise as covered in Chapters II and IV.

Then, the continuous-time LQG regulator is presented, based on a control law that employs state estimates from a Kalman filter:

$$\underline{u}^*(t) = -G_c^* \, \hat{\underline{x}}(t) \tag{A-2}$$

Next, the structure of a sampled-data, full-state feedback regulator is examined with a control law expressed as

$$\underline{u}^*(t_i) = -G_c^* \, \underline{x}(t_i) \tag{A-3}$$

Two forms of a sampled-data LQG regulator are examined. The first is based on the optimal control law given by

$$\underline{u}^*(t_i) = -G_c^* \, \hat{\underline{x}}(t_i^+) \tag{A-4}$$

The second is based on the suboptimal control law

$$\underline{u}^*(t_i) = -G_c^* \, \hat{\underline{x}}(t_i^-) \tag{A-5}$$

In the last two sections, the format is presented for a sampled-data PI controller designed on the basis of LQG methodology. First, the full-state feedback case is considered, then a Kalman filter is used to provide state estimates.

A.2 Continuous-time Controller Generic Structure

The generic format desired for the controllers considered in this thesis is given by

$$\underline{u}^*(t) = G_{cx} \, \underline{x}_c(t) + G_{cz} \, \underline{z}(t) + G_{cy} \, \underline{y}_d(t) \tag{A-6}$$

$$\underline{x}_c(t) = F_c \, \underline{x}_c(t) + B_{cz} \, \underline{z}(t) + B_{cy} \, \underline{y}_d(t) \tag{A-7}$$

which takes the form of an algebraic relation for the optimal control, $\underline{u}(t)$ and a propagation equation for the controller states. The controller states, $\underline{x}_c(t)$, are defined for the particular type of controller considered, and $\underline{z}(t)$ and $\underline{y}_d(t)$ are measurements and desired values of controlled variables, respectively.

For the full-state feedback case, perfect measurements of the states are available, i.e.,

$$\underline{z}(t) = \underline{x}(t) \tag{A-8}$$

and $\underline{y}_d(t) = \underline{0}$. Thus, it is seen by comparing Equations (A-1) and (A-6) that

$$G_{cx} = 0 \qquad \text{(A-9)}$$

$$G_{cz} = -G_c^* \qquad \text{(A-10)}$$

$$G_{cy} = 0 \qquad \text{(A-11)}$$

In this instance, there are no internal controller states, so Equation (A-7) is not maintained.

For an LQG regulator, $\underline{y}_d(t)$ is again zero, therefore G_{cy} and B_{cy} can be set to zero. The controller states are defined to be the conditional mean estimates from a Kalman filter, $\underline{\hat{x}}(t)$. By the certainty equivalence principle, $\underline{x}(t)$ in Equation (A-2) can be replaced by $\underline{\hat{x}}(t)$ which are now the controller states, $\underline{x}_c(t)$:

$$\underline{u}^*(t) = -G_c^* \, \underline{x}_c(t) \qquad \text{(A-12)}$$

Comparing Equations (A-6) and (A-12), it is seen that

$$G_{cz} = 0 \qquad \text{(A-13)}$$

$$G_{cz} = -G_c^* \qquad \text{(A-14)}$$

Recall that the continuous-time Kalman filter equation yields an estimate of the states, $\underline{\hat{x}}(t)$, where $\underline{\hat{x}}(t)$ is now defined as the controller states, $\underline{x}_c(t)$. This is given by

$$\dot{\underline{x}}_c(t) = F\,\underline{x}_c(t) + B\underline{u}^*(t) + K\left[\underline{z}(t) - H\,\underline{x}_c(t)\right] \qquad \text{(A-15)}$$

Substituting Equation (A-12) into the above and rearranging yields

$$\dot{\underline{x}}_c(t) = \left[F - BG_c^* - KH\right]\underline{x}_c(t) + K\,\underline{z}(t) \tag{A-16}$$

Comparing Equations (A-7) and (A-16), it is seen that

$$F_c = \left[F - BG_c^* - KH\right] \tag{A-17}$$

$$B_{cz} = K \tag{A-18}$$

A.3 Optimal Sampled-Data Generic Controller Structure

The generic structure of Equations (A-4) and (A-5) is given in sampled-data form by

$$\underline{u}^*(t_i) = G_{cx}\,\underline{x}_c(t_i) + G_{cz}\,\underline{z}(t_i) + G_{cy}\,\underline{y}_d(t_i) \tag{A-19}$$

$$\underline{x}_c(t_{i+1}) = \phi_c\,\underline{x}_c(t_i) + G_{cz}\,\underline{z}(t_i) + G_{cy}\,\underline{y}_d(t_i) \tag{A-20}$$

It is easily seen that for a sampled-data full-state feedback regulator, the generic form is identical to Equations (A-9) through (A-11).

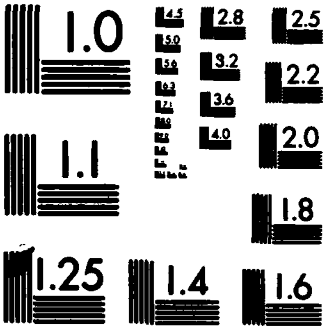
For the optimal LQG regulator, the actual state values are replaced by estimates just after measurements are taken at the sample times. Again, $\underline{y}_d(t)$ will be zero for a regulator, therefore G_{cy} and B_{cy} can be set to zero. Define the controller states to be $\hat{\underline{x}}(t_i^-)$, the state estimates just prior to a measurement update. The Kalman filter propagation and update relations for the estimates of the states are given by

$$\underline{x}(t_{i+1}^-) = \phi\underline{x}(t_i^+) + B_d\,\underline{u}^*(t_i) \tag{A-21}$$

AD-A138 425 ROBUST FLIGHT CONTROLLERS(U) AIR FORCE INST OF TECH 3/3
WRIGHT-PATTERSON AFB OH SCHOOL OF ENGINEERRNG
J M HOWEY DEC 83 AFIT/GAE/EE/83D-2

UNCLASSIFIED F/G 12/1 NL

END


1.0
4.5
5.0
5.6
6.3
7.1
8.0
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

$$\underline{\hat{x}}(t_i^{+}) = \left[I - KH \right] \underline{\hat{x}}(t_i^{-}) + K\underline{z}(t_i) \tag{A-22}$$

Replace $\underline{\hat{x}}(t_i^{-})$ by $\underline{x}_c(t_i)$ and substitute (A-22) into (A-4) to yield

$$\underline{U}^{*}(t_i) = -G_c^{*} \left[I-KH \right] \underline{x}_c(t_i) - G_c^{*}\ K\underline{z}(t_i) \tag{A-23}$$

Comparing (A-19) and (A-23), it is seen that

$$G_{cx} = -G_c^{*} \left[I-KH \right] \tag{A-24}$$

$$G_{cz} = -G_c^{*}\ K \tag{A-25}$$

Now substitute Equations (A-4) and (A-22) into (A-21), replacing $\underline{\hat{x}}(t_i-)$ with $\underline{x}_c(t_i)$:

$$\begin{aligned} \underline{x}_c(t_i+1) = & \left[\phi - B_d G_c^{*} \right] \left[I - KH \right] \underline{x}_c(t_i) \\ & + \left[\phi - B_d G_c^{*} \right] K\ \underline{z}(t_i) \end{aligned} \tag{A-26}$$

Comparing (A-20) and (A-26), it is seen that

$$\phi_c = \left[\phi - B_d G_c^{*} \right] \left[I-KH \right] \tag{A-27}$$

$$B_{cz} = \left[\phi - B_d G_c^{*} \right] K \tag{A-28}$$

A.4 Sub-optimal Sampled-Data Generic Controller Structure

For the suboptimal control law, where the state values are replaced by estimates just prior to the sample times, the controller states are defined to be $\underline{\hat{x}}(t_i^{-})$. The output, $\underline{y}_d(t_i)$, is zero, thus G_{cy} and B_{cy} can be set to zero. By direct comparison of Equations (A-5) and (A-29), it is seen that

$$G_{cx} = -G_c^* \qquad \text{(A-29)}$$

$$G_{cz} = 0 \qquad \text{(A-30)}$$

Now substitute Equation (A-5) and (A-22) into (A-21), replacing $\underline{\hat{x}}(t_i-)$ by $\underline{x}_c(t_i)$, to yield

$$\underline{x}_c(t_{i+1}) = \{\phi\left[I-KH\right] - B_d G_c^*\} \underline{x}_c(t_i)$$

$$+ \left[\phi K\right] \underline{z}(t_i) \qquad \text{(A-31)}$$

Thus, by comparing Equations (A-20) and (A-31), it is seen that

$$\phi_c = \{\phi\left[I-KH\right] - B_d G_c^*\} \qquad \text{(A-32)}$$

$$B_{cz} = \left[\phi K\right] \qquad \text{(A-33)}$$

A.5 Sampled-Data PI Conf.oller Generic Structure

An equivalent form for the PI control law derived in Chapter III which achieves Type-1 control is given by (Ref 1;23).

$$u^*(t_i) = -K_x \underline{x}(t_i) + K_z \underline{\xi}(t_i) + K_z \underline{y}_d(t_i) \qquad \text{(A-34)}$$

where $\underline{\xi}(t_i)$ are termed the pseudo-integral states and are expressed as

$$\underline{\xi}(t_{i+1}) = \underline{\xi}(t_i) + \left[\underline{y}_d(t_i) - \underline{y}(t_i)\right] \qquad \text{(A-35)}$$

For the PI controller, a non-zero output, $\underline{y}_d(t_i)$, will be allowed, where $\underline{y}(t_i)$, the actual output of the system, is given by

$$\underline{y}(t_i) = C\underline{x}(t_i) + D_y\, \underline{u}^*(t_i) \qquad \text{(A-36)}$$

For the full-state feedback case, the controller states are defined to be $\underline{\xi}(t_i)$. Substituting (A-35) and (A-36) into (A-34) and recalling ahat $\underline{x}(t_i) = \underline{z}(t_i)$ yields

$$\underline{u}^*(t_i) = K_z\underline{x}_c(t_i) - K_x\underline{z}(t_i) + K_z\underline{y}_d(t_i) \qquad \text{(A-37)}$$

Comparing this equation (A-19), it is seen that

$$G_{cx} = K_z \qquad \text{(A-38)}$$

$$G_{cz} = -K_x \qquad \text{(A-39)}$$

$$G_{cy} = K_z \qquad \text{(A-40)}$$

Then, by substituting Equations (A-34) and (A-36) into (A-35) with $\underline{x}_c(t_i) = \underline{\xi}(t_i)$:

$$\underline{x}_c(t_{i+1}) = \left[I-D_yK_z\right]\underline{x}_c(t_i) - \left[C-D_yK_x\right]\underline{z}(t_i)$$

$$+ \left[I-D_yK_z\right]\underline{y}_d(t_i) \qquad \text{(A-41)}$$

and comparing this to Equation (A-20) yields

$$\phi_c = \left[I-D_yK_z\right] \qquad \text{(A-42)}$$

$$B_{cz} = \left[C-D_yK_x\right] \qquad \text{(A-43)}$$

$$B_{cy} = \left[I-D_yK_z\right] \qquad \text{(A-44)}$$

With a Kalman filter in the loop, $\underline{x}(t_i)$ is replaced by $\hat{\underline{x}}(t_i^+)$ in Equations (A-34) and (A-36).

The controller states, $\underline{x}_c(t_i)$, for PI controller are defined to be an augmented vector containing the state estimates just prior to a measurement update and the pseudo-integral states:

$$\underline{x}_c(t_i) = \begin{bmatrix} \hat{\underline{x}}(t_i^-) \\ \\ \underline{\xi}(t_i) \end{bmatrix} \tag{A-45}$$

To generate the generic controller structure, first substitute the Kalman filter update relation (A-22) into Equations (A-21) and (A-34). This yields

$$\hat{\underline{x}}(t_{i+1}-) = \phi\{\left[I-KH\right]\underline{x}(t_i^-) + K\,\underline{z}(t_i)\} + B_d u^*(t_i) \tag{A-46}$$

$$\underline{u}^*(t_i) = -K_x\,\{\left[I-KH\right]\hat{\underline{x}}(t_i^-) + K\,\underline{z}(t_i)\} + K_z\,\underline{\xi}(t_i) \tag{A-47}$$

Substitute (A-47) into (A-46) to yield

$$\begin{aligned} \hat{x}(t_{i+1}^-) = {} & \left[\phi - B_dK_x\right]\left[I-KH\right]\hat{\underline{x}}(t_i^-) \\ & + B_dK_z\,\underline{\xi}(t_i) + \left[\phi - B_dK_x\right]K\,\underline{z}(t_i) \\ & + B_d\,K_z\,\underline{y}_d(t_i) \end{aligned} \tag{A-48}$$

Next, substitute Equations (A-22) and (A-36) into (A-35) to yield

$$
\begin{aligned}
\underline{\xi}(t_{i+1}) = {} & -\left[C - D_y K_x\right]\left[I-KH\right]\hat{\underline{x}}(t_i^-) \\
& + \left[I-D_y K_z\right]\underline{\xi}(t_i) - \left[C - D_y K_x\right] K\,\underline{z}(t_i) \\
& + \left[I-D_y K_z\right]\underline{y}_d(t_i)
\end{aligned}
\tag{A-49}
$$

Equations (A-47) and (A-48) in terms of the controller states, $\underline{x}_c(t_i)$, are given by

$$
\begin{aligned}
\underline{u}^*(t_i) = {} & \left[-K_x\{I-KH\} \;\vdots\; K_z\right]\underline{x}_c(t_i) \\
& - \left[K_x K\right]\underline{z}(t_i) + K_z\,\underline{y}_d(t_i)
\end{aligned}
\tag{A-50}
$$

$$
\begin{aligned}
\underline{x}_x(t_{i+1}) = {} & \left[\begin{array}{c:c} \left[\phi - B_d K_x\right]\left[I-KH\right] & B_d K_z \\ \hdashline -\left[C-D_y K_x\right]\left[I-KH\right] & \left[I-D_y K_z\right] \end{array}\right]\underline{x}_c(t_i) \\
& + \begin{bmatrix} \left[\phi - B_d K_x\right]K \\ -\left[C-D_y K_x\right]K \end{bmatrix}\underline{z}(t_i) + \begin{bmatrix} B_d K_z \\ \left[I-D_y K_z\right] \end{bmatrix}\underline{y}_d(t_i)
\end{aligned}
\tag{A-51}
$$

Comparing Equations (A-19) and (A-50), and (A-20) and (A-51), it is seen that

$$
G_{cx} = \left[-K_x\{I-KH\} \;\vdots\; K_z\right]
\tag{A-52}
$$

$$
G_{cz} = \left[-K_x K\right]
\tag{A-53}
$$

$$
G_{cy} = \left[K_z\right]
\tag{A-54}
$$

$$\phi_c = \left[\begin{array}{c:c} \left[\phi - B_d K_x\right]\left[I-KH\right] & B_d K_z \\ \hdashline -\left[C-D_y K_x\right]\left[I-KH\right] & \left[I-D_y K_z\right] \end{array}\right] \tag{A-55}$$

$$B_{cz} = \begin{bmatrix} \left[\phi - B_d K_x\right] K \\ \\ -\left[C-D_y K_x\right] K \end{bmatrix} \tag{A-56}$$

$$B_{cy} = \begin{bmatrix} B_d K_z \\ \\ \left[I-D_y K_z\right] \end{bmatrix} \tag{A-57}$$

APPENDIX B: Modifications and Additions to LQGRP

A.1 Introduction

This appendix is intended to be used as a supplement to Reference 21, Appendix D,which is the user's guide to a program entitled Linear Quadratic Gaussian Regulator Performance (LQGRP). The program provides the capability of designing and evaluating the performance of continuous-time and sampled-data LQG regulators. In addition, it contains options of enhancing the robustness of both types of regulators via the Doyle and Stein technique.

Three types of changes are made to the original program which are documented herein. The first type involves corrections to errors in the original source code. The second type entails modifications that result in more convenient input and output and a more efficient program. The third type of change generates additions so that the technique of injecting time-correlated noise into the design model while tuning the Kalman filter for robustness purposes can be applied to LQG regulators.

Corrections to the original source code (contained on subsequent pages) are bracketed and marked with a "C". Modifications are bracketed and marked with an "M", and additions are bracketed and marked with an "A".

A.2 Corrections to LQGRP

The first correction appears on line 6770. The variable name IRF2 replaces IRFM in the original code. IRFM is the dimension of the design model F matrix, but what should be passed to the subroutine is the dimension of F minus the number of deterministic states, which is IRF2.

Next, on line 15330, ICBM (the column dimension of the model B matrix) replaces IRFM in the second argument of the subroutine MAT3.

A third correction is made in lines 17120 and 17130. The original source code contains extra arguments in the call to subroutine DDTCON. They do not interfere with execution of the program, but they are not needed and have been deleted in this version.

In line 19430, BM (IDS, IDS) (starting address of the design model B matrix after deterministic states are deleted) replaces BM in the original program. It is intended to pass to the subroutine only the lower portion of the design model B matrix, BM, after the deterministic states (listed first) are deleted. The original program passes the entire B matrix. Making this correction changes the starting address of the array in subroutine DAS2 so that only the desired portion of BM is used.

Lines 19500 and 19510 are inserted into the program to correct an error in line 19520. Originally, RMD appears in place of WM4. RMD is the discrete-time measurement noise covariance matrix. However, the subroutine KFLTR requires a vector containing the diagonal elements of RMD. Thus, the diagonal elements of RMD are stored in the first column of WM4 in lines 15900 and 19510, and WM4 is passed to the subroutine instead of RMD. WM4 is a work-space array not originally being used at that point in the program.

The last correction is made to line 20840. Originally, the line read call MAT3, which calculates the transpose of the desired result. This is corrected by replacing it with CALL MAT3A.

B.3 Modifications to LQGRP

The matrices and vectors defined in lines 370 through 510 were originally of dimension 5 or 10. All matrices and vectors of dimension 5 are changed to 13, and those of dimension 10 are changed to 26. This change is reflected also in lines 680 through 700. This allows the number of states in the truth or design model to be a maximum number of 13.

The original version of LQGRP had 6 input/output options for entering and printing out truth and design model matrices. The version listed herein has 7. The first 2 options are unchanged. Option 3 now allows the user to change as many elements of a specified matrix as desired by listing the row, column and value of the element being changed. When the user enters a 0, the program exits this option. Options 4 and 5 are also unchanged. Option 6 initially zeros the entire array and then allows the user to enter as many elements as desired by entering the row, column and value of each element. Entering a 0 will exit this option. Option 7 takes the place of option 6 in the previous version. Thus, on a first run through the program, option 6 initializes the array to zero and then only non-zero elements need to be entered. On subsequent runs, changes can be made to any vector or matrix by using option 3.

The changes discussed above require that modifications be made to subroutines MVEC10 and MMAT10, which control the programs input and output. These are listed in lines 12370 to 12380, 12460 to 12530, 12590, 13060, 13130 to 13230 and 13280.

The program stores results of the performance evaluation routines to four data files for plotting. These data files contain the time histories of the following vectors and matrices, defined in Section 2.3:

$\underline{x}_a(t)$, $P_{x_a x_a}(t)$, $\underline{u}(t)$ and $P_{uu}(t)$. The modifications listed in lines 10650 and 10690 calculate and store the square roots of the diagonal elements (standard deviations) of $P_{x_a x_a}(t)$ and $P_{uu}(t)$ rather than storing the diagonal elements (variances). This change also occurs in lines 11570 and 11600. Notice in lines 11560 through 11660 that only the first 7 states of $\underline{x}_a(t)$ and the upper left (7x7) partition of $P_{x_a x_a}(t)$ is stored for plotting. This is an option of the user. Any portion of the vector and matrix may be stored by changing the 7 in lines 10630, 10770, 11550, 11610 and 11650 to the desired value or to IREM (the dimension of the design model F matrix) if the entire vector and matrix are desired.

The four vectors and matrices listed above are printed to the terminal during execution of the program. Currently, only the first 3 states of $\underline{x}_a(t)$ and the upper-left (3x3) partition of $P_{x_a x_a}(t)$ are printed. Any portion may be printed by changing the 3 in lines 10530 and 10570 to a desired value or to IRFM if the entire vector and matrix are desired. Note that all of $\underline{u}(t)$ and $P_{uu}(t)$ are stored and listed at the terminal.

The preceding modifications were made to LQGRP to make the program more convenient to use in conjunction with the problem considered in this thesis. None are necessary for execution of the program, as contrasted to the corrections of Section B.2.

B.4 Additions to LQGRP

This section describes the additions necessary to the program so that the technique of injecting colored noise into the design model during filter tuning may be applied. The changes are primarily lines of code inserted into the program. However, a few lines of the original source code are altered slightly, and these will be mentioned.

For a continuous-time system, the input prompt for colored noise is given in line 13870. If colored noise is not desired, the program skips to line 13920 and continues execution. Note that line 13920 is from the original program with the label 2900 inserted in front. If colored noise is desired, the dimension of the model F matrix, IRFM, is stored in a dummy variable, IHOLD, for later use. Then, in line 13910, subroutine CNOISE is called, which controls the augmentation of shaping filter states with the original design model.

Lines 13950 through 14020 augment zeros to the deterministic controller gain matrix, G_c^*, if the colored-noise options are executed. Lines 13950 and 13960 are from the original source code, altered by replacing IRFM with IHOLD. Recall, as discussed in Section 4.3, that G_c^* is calculated using the design model of the unaugmented system, and the dimensionality difference is taken care of by augmenting G_c^* with the appropriate number of columns of zeros.

For a sampled-data system, the colored noise input prompt should appear between lines 16710 and 16720. It is missing from this listing, but should read

```
WRITE (KOUT,*) 'COLORED INPUT NOISE (Y OR N)>'
```

Lines 16720 through 16760 then control whether the colored-noise routines are executed. Line 16760 is from the original program with a label 25 inserted.

On line 17090, IRFM is again replaced by IHOLD. Lines 17120 and 17130 are also from the original program with an additional argument, IHOLD, in the parameter list. Then, lines 17140 through 17190 check to see if the colored-noise routines were executed. If so, G_c^* is augmented with the appropriate number of columns of zeros.

In both the continuous-time and sampled-data case, it is desired to use the original design model for controller gain calculations, as mentioned above, in subroutines CDTCON and DDTCON, which calculate deterministic controller gains for the continuous-time and sampled-data case, respectively. These subroutines are essentially unchanged except that IHOLD is substituted for IRFM where indicated in CDTCON (lines 5110 through 5890). In DDTCON, IHOLD is an additional parameter in the argument list and replaces IRFM where indicated (lines 20480 through 21030).

Lines 24770 through 24930 contain subroutine CNOISE which controls what type of shaping filters are augmented to the design model. An input prompt asks if a first- or second-order shaping filter is desired. Subroutine FORDER is called (lines 24940 through 25510) if a first-order shaping filter is desired. If a second-order filter is desired, then subroutine SORDER is called (lines 25520 through 26160). Both subroutines prompt for the shaping filter design parameters described in Chapter IV. Then, the design model is augmented with the shaping filter states and appropriate partitions of the augmented matrices are zeroed. Finally, the dimensions of the design model F and G matrices are altered to reflect the higher dimension after augmenting.

The matrices described above thus modify LQGRP so that shaping filter states can be augmented with the original system design model. It should be noted that the program will still ask if modification via the Doyle and Stein technique is desired if the colored-noise options are chosen. Either the colored-noise options or the Doyle and Stein options may be chosen, but not both.

Additionally, it is desirable to have the original design and truth models stored on Tape 8 (Ref 21) so that they can be read into the program via input option 18. When the colored-noise options are chosen, the design model is altered. Thus, to make subsequent runs through LQGRP, the original design and truth models must be read back in.

```
      PROGRAM LQGRP(OUTPUT=64,TAPE6=OUTPUT,TAPE12=64,TAPE13=64,
     1TAPE14=64,TAPE15=64,INPUT=64,TAPE10=INPUT,TAPE3=64,TAPE7)
C  THIS PROGRAM PERFORMS A PERFORMANCE ANALYSIS FOR THE LINEAR QUADRATIC
C  GAUSSIAN CONTROLLERS DESCRIBED BY D(X)/DT=FX+BU+GW AND A QUADRATIC
C  COST FUNCTION,       IF DIFFERENT CONTROL ALGORITHMS ARE SUPPLIED THE
C PROGRAM WILL STILL DO A PERFORMANCE ANALYSIS.  THE METOHODOLOGY IS
C  BASED ON THE PERF ANAL. SECTION IN CHAFTER  14 OF P. S. MAYBECKS
C  TO BE PUBLISHED VOLUME 2 OF STOCH,MODELS,EST AND CONTROL
C
C  MANY OF THE SUBROUTINES USED FOR MATRIX MANIPULATION COME FROM THE
C  ROUTINES  COMPILED BY D. KLIENMAN(TR-75-4, ONR CONTRACT #
C  N00014-75-C1367)
C
C  IN THE FOLLOWING PROGRAM TRUTH MODEL MATRICES ARE TWO LETTERS WITH
C  THE LAST LETTER BEING  -T- .  CONTROLLER MODEL MATRICES ARE TWO
C  LETTERS ENDING IN -M-.  -T- FOLLOWING APARTICULAR MATRIX NAME
C  INDICATES THE MATRIX IS TRANSPOSED.  -I- FOLLOWING A PARTICULAR
C  MATRIX NAME INDICATES THE INVERSE OF THE MATRIX
C
C
C  COMMON BLOCKS MAIN1,MAIN2,INOU ARE REQUIRED BY THE KLIEMAN ROUTINES
C
C****** KLIENMAN ROUTINE -GATIO- REQUIRES A BLANK CARD TO END THE READ**
C
C-----INPUT FROM TAPE10     ----OUTPUT TO TAPE11  .

      CHARACTER MSG*60,DSCRPT*60
      REAL FT(13,13),BT(13,13),GT(13,13),HT(13,13),FM(13,13),
     1   FM(13,13),BM(13,13),GM(13,13),XO(13),HM(13,13),QM(13,13),
     1PO(13,13),QT(13,13),RT(13,13),COM1(26,26),COM2(26,26),GCSTR(13,13
     1),WXU(13,13),RKFSS(13,13),WV1(13),WV2(13),
     1WM1(13,13),WM2(13,13),WM3(13,13),WM4(13,13),WM5(13,13),WM6(13,13),
     1WMA(26,26),WMB(26,26),WMC(26,26),WMD(26,26),WME(26,26),WMF(26,26),
     1   WUU(13,13),WXX(13,13),FA(26,26),BA(26,26),GA(26,26),MXA(26),      M
     1   GCX(13,13),GCZ(13,13),QA(26,26),PXA(26,26),PXVA(26,26),
     1   MUOUT(13),MXTOUT(13),PXTOUT(13),PUOUT(13),YO(1000),
     1   GUA(26,26),MXAMIN(13),MXAMAX(13),MUMIN(13),MUMAX(13),
     1  PXTMIN(13),PXTMAX(13),PUMIN(13),PUMAX(13),GCZA(26,26),
     1 RA(26,26),BCY(13,13),BCZ(13,13),GCY(13,13),FC(13,13),
     1WM7(13,13),WM8(13,13),WM9(13,13),WM10(13,13),WV3(26),WV4(26)
      INTEGER IFLGCZ,IRY
      REAL MU(13)
      COMMON /RNTIM/ RNTIME,DELTIM
      COMMON /MAIN2/COM2
      COMMON /MAIN1/NDIM,NDIM1,COM1
      COMMON / INOU/ KIN,KOUT,KPUNCH
      COMMON /MAIN4/NDIM2,NDIM3
      COMMON /MAUN5/ MSG
      COMMON /MAIN6/ ICBT,ICBM,ICFA,ICGA,ICGM,ICGT,ICQA,IRFA,IRFM,IRFT,
     1   IRMT,IRQA,IO,LOG,IFMM,NUMOTS
C
C
C
C
C
      KIN=10
      KOUT=6
      KPUNCH=7
      NDIM=13                                                              M
      NDIM1=NDIM+1
      NDIM2=26
      NDIM3=1000
C
```

FOLLOWING

Reproduced from best available copy.

PAGES

```
C
C
C    **********MAIN PROGRAM FOLLOWS *******
C
C----   IRXX, ICXX INDICATE  ROWS,COLUMNS OF MATRIX XX
C
C*****THIS PROGRAM CAN HANDLE UP TO 1000 DIFFERENT COMBINATIOS OF
C          RUNTIME,DELTIM,AND UNSPECIFIED PARAMETERS
      DO 2932 LOG=1,1000
      WRITE(KOUT,11)' '
11    FORMAT(A1,/)
12     FORMAT(A1)
      WRITE(KOUT,*)'THIS IS RUN NUMBER ',LOG
      WRITE(KOUT,*)' '
      WRITE(KOUT,*)' '
      WRITE(KOUT,*)'ENTER A DESCRIPTION OF THIS RUN>'
      WRITE(KOUT,*)' '
      READ(KIN,12)DSCRPT
      WRITE(KOUT,*)' '
      WRITE(KOUT,*)' '
      CALL INPUTM(FT,BT,GT,HT,FM,BM,GM,HM,PO,QT,QM,RT,RM,XO,
     1  WUU,WXX,WXU)
      IF (IO.EQ.0) THEN
      GO TC 2933
      END IF
      CALL RGS(GCSTR,RKFSS,GCX,GCY,GCZ,BCY,BCZ,FC,YD,
     1WM1,WM2,WM3,WM4,WM5,WM6,WM7,WM8,WM9,WM10,WV1,WV2,
     1WMA,WMB,WMC,WMD,WME,WMF,FT,BT,GT,HT,QT,RT,
     1 FM,BM,GM,HM,QM,RM,XO,PC,WXX,WUU,WXU,FA,BA,GA,QA,GUA,
     1 MXA,PXA,RA,PXVA,GCZA,IRY,IFLGCZ,IFLGSD,
     1MU,PUMAX,PUMIN,PXTMAX,PXTMIN,MUMAX,MUMIN,MXAMAX,MXAMIN,
     1 PUCUT,PXTOUT,MXTOUT,MUCUT,WV3,WV4)
      WRITE(KOUT,*)' '
      WRITE(KOUT,*)'DO YOU WISH TO CALCULATE THE EIGENVALUES OF THE CLOS
     1ED-LOOP STATE TRANSITION MATRIX USED IN THE PERFORMANCE ANALYSIS
     1(APPLICABLE TO BOTH THE CONTINUCUS-TIME AND SAMPLED-DATA CASE)
     1  Y  OR  N>'
      READ(KIN,12)MSG
      IF (MSG.EQ.'Y') THEN
      WRITE(KOUT,*)'THE CLOSED-LOOP STATE TRANSITION MATRIX EIGENVALUES
     1 ARE...'
      NSAV=NOIM
      NOIM=NOIM2
      NOIM1=NOIM2+1
      CALL MEIGN(WMA,WV3,WV4,IRFA,WME)
      NOIM=NSAV
      NOIM1=NSAV+1
      END IF
      WRITE(KOUT,*)' '
      WRITE(KOUT,*)'TYPE Y  TO   PERFORM THE COVARIANCE ANALYSIS,  TYPE
     1 N TO SKIP IT>'
      READ(KIN,12)MSG
      IF (MSG.EQ.'N')THEN
      GO TO 2932
      END IF
      CALL PERFAL( IRY,IFLGCZ,MXA,GCY,GUA,PXA,PXVA,IFLGSD,
     1 RA,GCZA,YD,WMA,WMB,WME,WMF,WMD,WM1,MUOUT,MXTOUT,PUCUT,PXTOUT,
     1  MXAMIN,MXAMAX,PXTMIN,PXTMAX,MUMIN,MUMAX,PUMIN,PUMAX,MU,WMC,
     1 WV3,WV4)
2932  CONTINUE
2933  WRITE(KOUT,*)'PROGRAM TERMINATED, NO MORE INPUT DATA'
      END
* DECK STORED
      SUBROUTINE STORED(IWCHRN,RNTIME,DELTIM,IFSTCL,IS12SZ,
     1   IS13SZ,IS14SZ,IS15SZ,STOR12,STOR13,STOR14,STOR15,NOIMS)
C      STORE DATA TO PERMANENT FILE
```

```
CTHIS SUBROUTINE STORES PLOT DATA TO LOCAL FILES.  TAPE12,TAPE13.
C        TAPE14- TAPE15 FOR PERMANENT STORAGE.  ISXXSZ IS THE SIZE OF THE
C        DATA ARRAYS,STORXX.  NDIM IS THE CALLING PROGRAM DIMENSION OF
C        THE ARRAYS.      NOTE THAT   RUNTIME*DELTIM+2  GIVES THE TOTAL
C        NUMBER OF DATA POINTS IN THE RUN.  THE LAST ENTRY IN EACH
C        DATA FILE IS THE SCALE FACTOR FOR THE DATA ON THE FILE.  THE
C        NEXT TO LAST ENTRY IS THE MINIMUM VALUE OF THE DATA IN THE FIL
      DIMENSION STOR12(NDIM6),STOR13(NDIM6),STOR14(NDIM6),STOR15(NDIM6)
      COMMON /INOU/ KIN,KOUT,KPUNCH
      IF (IFSTCL.EQ.0) THEN
C     PUT A RUN HEADER ON THE TAPE
      OPEN(UNIT=12,ERR=10,FILE='TAPE12',RECL=80)
      OPEN(UNIT=13,ERR=10,FILE='TAPE13',RECL=80)
      OPEN(UNIT=14,ERR=10,FILE='TAPE14',RECL=80)
      OPEN(UNIT=15,ERR=10,FILE='TAPE15',RECL=80)
      WRITE(12,101) IWCHRN,RNTIME,DELTIM,IS12SZ
      WRITE(13,101) IWCHRN,RNTIME,DELTIM,IS13SZ
      WRITE(14,101) IWCHRN,RNTIME,DELTIM,IS14SZ
      WRITE(15,101) IWCHRN,RNTIME,DELTIM,IS15SZ
 101  FORMAT(' ',I10,1X,2E19.6,I10)
      END IF
      IF (IFSTCL.LT.0) THEN
C    CLOSE FILES AND PUT END OF FILE MARKER ON THEM.
      CLOSE(12,ERR=10)
      CLOSE(13,ERR=10)
      CLOSE(14,ERR=10)
      CLOSE(15,ERR=10)
      RETURN
      END IF
      WRITE(12,102)(STOR12(I),I=1,IS12SZ)
      WRITE(13,102)(STOR13(I),I=1,IS13SZ)
      WRITE(14,102)(STOR14(I),I=1,IS14SZ)
      WRITE(15,102)(STOR15(I),I=1,IS15SZ)
      RETURN
 10   WRITE(KOUT,*)'AN ERROR HAS OCCURRED IN THE STORED ROUTINE'
 102  FORMAT(3(4(' ',E19.6,1),/))
      END
* DECK INPUTM
      SUBROUTINE INPUTM(FT,BT,GT,HT,FM,BM,GM,HM,PO,QT,QM,RT,RM,XO,
     1WUU,WXX,WXU)
      CHARACTER MSG*60,MSG1*50
      REAL FT(NDIM,NDIM),BT(NDIM,NDIM),GT(NDIM,NDIM),HT(NDIM,NDIM),
     1HM(NDIM,NDIM),
     1QM(NDIM,NDIM),RM(NDIM,NDIM),XO(NDIM),
     1   WUU(NDIM,NDIM),WXX(NDIM,NDIM),FM(NDIM,NDIM),BM(NDIM,NDIM),
     1 GM(NDIM,NDIM),WXU(NDIM,NDIM),
     1 PO(NDIM,NDIM),QT(NDIM,NDIM),RT(NDIM,NDIM),COM1(1),COM2(1)
      COMMON /MAIN4/NDIM2,NDIM3
      COMMON /MAIN2/COM2
      COMMON /MAIN1/NDIM,NDIM1,COM1
      COMMON / INOU/ KIN,KOUT,KPUNCH
      COMMON /MAUN5/ MSG
      COMMON /MAIN6/ ICBT,ICBM,ICFA,ICGA,ICGM,ICGT,ICQA,IRFA,IRFM,IRFT,
     1   IRHT,IRQA,IO,LOG,IRHM,NUPOTS
      MSG1='*****-----*****-----*****-----*****-----*****'
C        ICFT=IRFT,ICFM=IRFM,IRBT=IRFT,IRBM=IRFM,IRGT=IRFT,IRGM=IRFM
C        IRX=IRF,IRQ=ICQ=ICGM,IRR=ICR=IRHT,IRWXX=ICWXX=IRFM,
C        IRWUU=ICWUU=ICBM,ICHT=IRFT,ICHM=IRFM,IRGCZ=ICBT,ICGCZ=IRHT,
C    IO IS A INPUT ROUTINE PARAMETER--1=READ,2=READ&PRINT,
C    3= PRINT ONLY, 4=PUNCH
C
      NSAV=NDIM1
      NDIM1=NDIM
      IF (LOG.EQ.1) THEN
      WRITE(KOUT,*) MSG1
      WRITE(KOUT,*)'THE I/O OPTIONS ARE 0,1,2,3,4,5,6......'
      WRITE(KOUT,*)'1-READ ENTIRE ARRAY/VECTOR,2-READ AND PRINT'
      WRITE(KOUT,*)'ENTIRE ARRAY/VECTOR, 3-READ,AND 4-READ AND PRINT'
      WRITE(KOUT,*)'SELECTED ARRAY/VECTOR ELEMENTS, 5 PRINT ENTIRE'
      WRITE(KOUT,*)'ARRAY/VECTOR.  6 TO PRE-ZERO ARRAY ELEMENTS THEN'     M
      WRITE(KOUT,*)'ENTER SELECTED ELEMENTS.  7 OR GREATER IF NO'
      WRITE(KOUT,*)' MORE INPUT.'
      WRITE(KOUT,*)MSG1
      WRITE(KOUT,*)'SELECT WHICH MATRIX YOU WISH TO ENTER.'
      WRITE(KOUT,*)'BY ENTERING THE APPROPRIATE NUMBER. 1-FT,2-BT'
      WRITE(KOUT,*)'3-GT,4-HT,5-FM,6-BM,7-GM,8-HM,9-PO,10-QT,11-RT,>'
      WRITE(KOUT,*)'12-QM,13-RM,14-XO,15-WUU,16-WXX,17-WXU,18-EQUATE ALL
     1'
      WRITE(KOUT,*)'CONTROLLER MODEL MATICES TO THEIR '
      WRITE(KOUT,*)'TRUTH MODEL COUNTERPARTS,19----- NO MORE DATA'
      WRITE(KOUT,*)'ENTRIES TO BE MADE. 20--STORE ALL MATRICES ON TAPE7'
      WRITE(KOUT,*)'21-  READ ALL MATRICES FROM TAPE8'
      WRITE(KOUT,*)MSG1
      WRITE(KOUT,*)MSG1
      WRITE(KOUT,*)' '
      WRITE(KOUT,*)'FOR SAMPLED DATA MEASUREMENTS,ENTER EITHER A CONTINU
     1OUS RM  TO USE TO APPROXIMATE THE DISCRETE TIME RMD(RMD=RM/SAMPLE
     1TIME) OR ENTER THE DISCRETE TIME RMD'
      WRITE(KOUT,*)MSG1
      WRITE(KOUT,*)MSG1
      END IF
      DO 992 INPUT=1,1000
      WRITE(KOUT,33333)' '
33333 FORMAT(A10,/)
      WRITE(KOUT,*)'ENTER CODE FOR WHICH ARRAY/VECTOR TO BE INPUT>'
      READ(KIN,*)IWCHMA
      GO TO(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21)IWCHMA
C
C***TRUTH MODEL INPUT
1     WRITE(KOUT,*)'ENTER-I/O OPTION, FT MATRIX SIZE>'
      READ(KIN,*,END=27)IO,IRFT
      MSG='TRUTH MODEL F MATRIX ENTRIES'
      CALL MMATIO (FT,IRFT,IRFT,IC,KIN,KOUT,NDIM,NDIM1)
      IF (IO.EQ.0) THEN
      RETURN
      END IF
      GO TO 992
2     WRITE(KOUT,*)'ENTER-I/O OPTIONS, COLUMN SIZE OF BT>'
      READ(KIN,*,END=27)IO,ICBT
      MSG='TRUTH MODEL  B  MATRIX ENTRIES'
      CALL MMATIO (BT,IRFT,ICBT,IC,KIN,KOUT,NDIM,NDIM1)
      IF (IO.EQ.0) THEN
      RETURN
      END IF
       GO TO 992
3     WRITE(KOUT,*)'ENTER-I/O OPTION, COLUMN SIZE OF GT>'
      READ(KIN,*,END=27)IO,ICGT
      MSG='TRUTH MODEL G MATRIX ENTRIES'
      CALL MMATIO (GT,IRFT,ICGT,IC,KIN,KOUT,NDIM,NDIM1)
      IF (IO.EQ.0) THEN
      RETURN
      END IF
      GO TO 992
4     WRITE(KOUT,*)'ENTER- I/O OPTION, ROW SIZE OF HT>'
      READ(KIN,*,END=27)IO,IRHT
      MSG='H MATRIX ENTRIES'
      CALL MMATIO (HT,IRHT,IRFT,IC,KIN,KOUT,NDIM,NDIM1)
      IF (IO.EQ.0) THEN
      RETURN
      END IF
      GO TO 992
C***INPUT CONTROLLER MODEL
```

```
5     WRITE(KOUT,*)'ENTER-I/O OPTION, FM MATRIX SIZE>'
      READ(KIN,*,END=27)IO,IRFM
      MSG='CONTROLLER MODEL F MATRIX ENTRIES'
      WRITE(KOUT,*)'ENTER THE NUMBER OF DETERMINISTIC STATES IN THIS MOD
     1EL>'
      READ(KIN,*,END=27)NUMDTS
      CALL MMATIO (FM,IRFM,IRFM,IO,KIN,KOUT,NDIM,NDIM1)
      IF (IO.EQ.0) THEN
      RETURN
      END IF
      GO TO 982
6      WRITE(KOUT,*)'ENTER-I/O OPTION, COLUMN SIZE OF BM>'
      READ(KIN,*,END=27)IO,ICBM
      MSG='CONTORLLER MODEL B MATRIX ENTRIES'
      CALL MMATIO (BM,IRFM,ICBM,IO,KIN,KOUT,NDIM,NDIM1)
      IF (IO.EQ.0) THEN
      RETURN
      END IF
      GO TO 982
7     WRITE(KOUT,*)'ENTER-I/O OPTION, COLUMN SIZE OF GM>'
      READ(KIN,*,END=27)IO,ICGM
      MSG='CONTROLLER MODEL G MATRIX ENTRIES'
      CALL MMATIO (GM,IRFM,ICGM,IO,KIN,KOUT,NDIM,NDIM1)
      IF (IO.EQ.0) THEN
      RETURN
      END IF
      GO TO 982
8     WRITE(KOUT,*)'ENTER I/O OPTION,ROW SIZE OF HM>'
      READ(KIN,*,END=27)IO,IRHM
      MSG='THE CONTROLLER MODEL MEASUREMENT MATRIX, HM, IS'
      CALL MMATIO(HM,IRHM,IRFM,IO,KIN,KOUT,NDIM,NDIM1)
      IF (IO.EQ.0) THEN
      RETURN
      ENDIF
      GO TO 982
9     WRITE(KOUT,*)'FT MUST BE ENTERED THRU OPTION 1 PRIOR TO USING THIS
     1 OPTION.DO YOU WISH TO ABORT THIS OPTION, Y  OR  N>'
      READ(KIN,97,END=27)MSG
      IF (MSG.EQ.'Y') THEN
      GO TO 982
      END IF
      WRITE(KOUT,*)'ENTER I/O OPTION,IRFT IS ASSUMED SIZE OF PO>'
      READ(KIN,*,END=27)IO
      MSG='THE INITIAL COVARIANCE MATRIX,PO, IS'
      CALL MMATIO (PO,IRFT,IRFT,IO,KIN,KOUT,NDIM,NDIM1)
      IF (IO.EQ.0) THEN
      RETURN
      END IF
      GO TO 982
10     WRITE(KOUT,*)'ENTER I/O OPTION, ICGT IS ASSUMED SIZE OF QT>'
      READ(KIN,*,END=27)IO
      MSG='THE INPUT NOISE STRENGTH MATRIX QT IS'
      CALL MMATIO (QT,ICGT,ICGT,IO,KIN,KOUT,NDIM,NDIM1)
      IF (IO.EQ.0) THEN
      RETURN
      END IF
      GO TO 982
11      WRITE(KOUT,*)'ENTER I/O OPTION,IRHT IS ASSUMED SIZE OF RT>'
      READ(KIN,*,END=27)IO
      MSG='THE MEASUREMENT NOISE STRENGTH MATRIX RT IS'
      CALL MMATIO (RT,IRHT,IRHT,IO,KIN,KOUT,NDIM,NDIM1)
      IF (IO.EQ.0) THEN
      RETUEN
      END IF
      GO TO 982
12    WRITE(KOUT,*)'ENTER I/O OPTION,ICGM IS ASSUMED SIZE OF QM>'
```

```
      READ(KIN,*,END=27)IO
      MSG='CONTROLLER MODEL INPUT NOISE STRENGTH MATRIX, QM'
      CALL MMATIO(QM,IQGM,IQGM,IO,KIN,KOUT,NDIM,NDIM1)
      IF (IO.EQ.0) THEN
      RETURN
      END IF
      GO TO 982
 13   WRITE(KOUT,*)'ENTER I/O OPTION, IRHM IS ASSUMED SIZE OF RM>'
      READ(KIN,*,END=27)IO
      MSG='CONTROLLER MODEL MEASUREMENT NOISE STRENGHT MATRIX, RM'
      CALL MMATIO(RM,IRHM,IRHM,IO,KIN,KOUT,NDIM,NDIM1)
      IF (IO.EQ.0) THEN
      RETURN
      END IF
      GO TO 982
 14   WRITE(KOUT,*)'FT MUST BE ENTERED THRU OPTION 1 PRIOR TO USING THIS
     1 OPTION.  DO YOU WISH TO ABORT THIS OPTION, Y  OR  N>'
      READ(KIN,97,END=27)MSG
      IF (MSG.EQ.'Y') THEN
      GO TO 982
      END IF
        WRITE(KOUT,*)'ENTER I/O OPTION,IRFT IS ASSUMED SIZEOF XO>'
      READ(KIN,*,END=27)IO
      MSG='THE INITIAL STATE VECTOR, XO, IS'
      CALL MVECIO (XO,IRFT,IO,KIN,KOUT,NDIM)
      IF (IO.EQ.0) THEN
      RETURN
      END IF
      GO TO 982
 15   WRITE(KOUT,*)'GM MUST BE ENTERED THRU OPTION 6 OR 17 PRIOR TO USIN
     1G THIS OPTION.  DO YOU WISH TO ABORT THIS OPTION,  Y  OR  N>'
      READ(KIN,97,END=27)MSG
      IF (MSG.EQ.'Y') THEN
      GO TO 982
      END IF
       WRITE(KOUT,*)'ENTER I/O OPTION,ICGM IS ASSUMED SIZE OF WUU>'
      READ(KIN,*,END=27)IO
      MSG='THE CONTROL FUNCTION COST WEIGHTING MATRIX, WUU'
      CALL MMATIO(WUU,ICGM,ICGM,IO,KIN,KOUT,NDIM,NDIM1)
      IF (IO.EQ.0) THEN
      RETURN
      END IF
      GO TO982
 16   WRITE(KOUT,*)'FM MUST BE ENTERED THRU OPTION 5 OR 17 PRIOR TO USIN
     1G THIS OPTION.  DO YOU WISH TO ABORT THIS OPTION, Y  OR N>'
      READ(KIN,97,END=27)MSG
      IF (MSG.EQ.'Y') THEN
      GO TO 982
      END IF
      WRITE(KOUT,*)'ENTER I/O OPTION,IRFM IS ASSUMED SIZE OF WXX>'
      READ(KIN,*,END=27)IO
      MSG='THE STATE COST WEIGHTING MATRIX, WXX'
      CALL MMATIO(WXX,IRFM,IRFM,IO,KIN,KOUT,NDIM,NDIM1)
      IF (IO.EQ.0) THEN
      RETURN
      END IF
      GO TO 982
 18   WRITE(KOUT,*)'ALL CONTROLLER MODEL MATRICES HAVE BEEN SET EQUAL TO
     1 THEIR TRUTH MODEL COUNTERPARTS'
      WRITE(KOUT,*)'FT,BT,GT,HT,QT,RT MUST BE ENTERED PRIOR TO USING THI
     1S OPTION.  THE NUMBER OF DETERMINISTIC STATES  MUST BE ENTERED IN
     1 THIS OPTION(FOR CONTROLLER MODEL).'
      WRITE(KOUT,*)' DO YOU WISH TO ABORT THIS OPTION,Y OR N>'
      READ(KIN,97,END=27)MSG
      IF (MSG.EQ.'Y') THEN
      GO TO 982
      END IF
      WRITE(KOUT,*)'ENTER THE NUMBER OF DETERMINISTIC STATES IN THE CONT
     1ROLLER MODEL>'
      READ(KIN,*,END=27)NUMDTS
      IRFM=IRFT
 97   FORMAT(A1)
      ICBM=ICBT
      ICGM=ICGT
      IRHM=IRHT
      CALL EQUATE(FM,FT,IRFT,IRFT)
      CALL EQUATE(BM,BT,IRFT,ICBT)
      CALL EQUATE(GM,GT,IRFT,ICGT)
      CALL EQUATE(HM,HT,IRHT,IRFT)
      CALL EQUATE(QM,QT,ICGT,ICGT)
      CALL EQUATE(RM,RT,IRHT,IRHT)
      GO TO 982
 17   WRITE(KOUT,*)'NOTE THAT FM AND BM MUST BE ENTERED THROUGH APPROPRI
     1ATE OPTIONS PRIOR TO EXECUTING THIS OPTION. DO YOU WISH TO ABORT T
     1HIS OPTION, Y  OR  N>'
      READ(KIN,97,END=27)MSG
      IF (MSG.EQ.'Y')THEN
      GO TO 982
      END IF
      WRITE(KOUT,*)'ENTER I/O OPTION, IRFM X ICBM ASSSUMED SIZE WXU>'
      READ(KIN,*,END=27)IO
      MSG='THE CROSS (STATE-CONTROL) COST WEIGHTING MATRIX, WXU'
      CALL MMATIO(WXU,IRFM,ICBM,IO,KIN,KOUT,NDIM,NDIM1)
      IF (IO.EQ.0)THEN
      RETURN
      END IF
      GO TO 982
 20   WRITE(KOUT,*)' THIS OPTION STORES ALL MATRICES ON TO TAPE7, DO YOU
     1 WISH TO ABORT THIS OPTION,Y  OR N>'
      READ(KIN,97)MSG
         IF (MSG.EQ.'Y')THEN
         GO TO 982
          END IF
      WRITE(7,*)IRFT,ICBT,ICGT,IRHT,IRFM,ICBM,ICGM,IRHM,NUMDTS
      WRITE(7,*)((FT(I,J),J=1,IRFT),I=1,IRFT),((BT(I,J),J=1,ICBT),I=1,IR
     1FT),((GT(I,J),J=1,ICGT),I=1,IRFT),((HT(I,J),J=1,IRFT),I=1,IRHT),
     1((FM(I,J),J=1,IRFM),I=1,IRFM),((BM(I,J),J=1,ICBM),I=1,IRFM)
      WRITE(7,*)((GM(I,J),J=1,ICGM),I=1,IRFM),((HM(I,J),J=1,IRFM),I=1,IR
     1HM),((PO(I,J),J=1,IRFT),I=1,IRFT),(XO(I),I=1,IRFT),((WUU(I,J),
     1 J=1,ICBM),I=1,ICBM),((WXX(I,J),J=1,IRFM),I=1,IRFM)
      WRITE(7,*)((QT(I,J),J=1,ICGT),I=1,ICGT),((RT(I,J),J=1,IRHT),I=1,IR
     1HT),((QM(I,J),J=1,ICGM),I=1,ICGM),((RM(I,J),J=1,IRHM),I=1,IRHM)
      WRITE(7,*)((WXU(I,J),J=1,ICBM),I=1,IRFM)
      GO TO 982
 21     WRITE(KOUT,*)' THIS OPTION READS ALL MATRICES FROM TAPE8, DO YOU
     1 WISH TO ABORT THIS OPTION,  Y  OR  N>'
      READ(KIN,97)MSG
      IF (MSG.EQ.'Y') THEN
           GO TO 982
      END IF
      WRITE(KOUT,*)'DO YOU WISH TO REWIND TAPE8 BEFORE THE READ,Y OR N>'
      READ(KIN,97)MSG
      IF (MSG.EQ.'Y') THEN
         REWIND(8)
      END IF
      READ(8,*,END=10033)IRFT,ICBT,ICGT,IRHT,IRFM,ICBM,ICGM,IRHM,NUMDTS
      READ(8,*,END=10033)((FT(I,J),J=1,IRFT),I=1,IRFT),((BT(I,J),J=1,ICB
     1T),I=1,IRFT),((GT(I,J),J=1,ICGT),I=1,IRFT),((HT(I,J),J=1,IRFT),I=1
     1,IRHT),((FM(I,J),J=1,IRFM),I=1,IRFM),((BM(I,J),J=1,ICBM),I=1,IRFM)
      READ(8,*,END=10033)((GM(I,J),J=1,ICGM),I=1,IRFM),((HM(I,J),J=1,IRF
     1M),I=1,IRHM),((PO(I,J),J=1,IRFT),I=1,IRFT),(XO(I),I=1,IRFT),((WUU(
     1 I,J),J=1,ICBM),I=1,ICBM),((WXX(I,J),J=1,IRFM),I=1,IRFM)
```

```
      READ(5,*,END=10333)((DT(I,J),J=1,IDGT),I=1,IDGT),((HT(I,J),J=1,IRH
     1T),I=1,IRHT),((DM(I,J),J=1,IDGM),I=1,IDGM),((FM(I,J),J=1,IRMM),
     1 I=1,IRMM)
      READ(5,*)((WAU(I,J),J=1,ICBM),I=1,I-FM)
      GO TO 982
10333 WRITE(KOUT,*)'END OF. FILE ENCOUNTERED DURING READ OF TAPE5'
982      CONTINUE
19        CONTINUE
      NOIM1=NSAV
      RETURN
27    IO=0
C
      END
* DECK MEIGN
      SUBROUTINE MEIGN(A,AREV,AIEV,CURSZA,WM1)
      CHARACTER MSG*60
      DIMENSIONA(NOIM,NOIM),AREV(NOIM),AIEV(NOIM),WM1(NOIM,NOIM)
      DIMENSION COM1(1),COM2(1)
      COMMON /MAIN1/NOIM,NOIM1,COM1
      COMMON / INOU/ KIN,KOUT,KPUNCH
      COMMON /MAIN2/ COM2
      COMMON /MAUN5/ MSG
C   THE  CALLING ROUTINE MUST SUPPLY A WORKING MATRX NOIM X NOIM --WM1
C***FIND THE EIGENVALUES OF A  ,NR=0 TELLS THE ROUTINE TO CALCULATE
C       EIGENVALUES ONLY
C  NOIM MUST BE THE DIMENSION OF A IN THE CALLING PROGRAM
      NR=0
      NR=0
      C1=1.0
      CALL IDNT(CURSZA,WM1,C1)
C WM1 =I   CURSZA X CURSZA
      CALL EIGEN(CURSZA,A,AREV,AIEV,WM1,NR)
      IO=5
      NSAV=NOIM1
      NOIM1=NOIM
      MSG='REAL PARTS OF THE EIGENVALUES '
      CALL MVECIO (AREV,CURSZA,IO,KIN,KOUT,NOIM)
      MSG='IMAG PARTS OF THE EIGENVALUES '
      CALL MVECIO (AIEV,CURSZA,IO,KIN,KCUT,NOIM)
C
      NOIM1=NSAV
      END
* DECK COTCON
      SUBROUTINE COTCON(FM,BM,WXX,WUU,GCSTR,IHOLD,ICSM,WM1,WM2,           A
     1WM3,WM4,WM5,WM6,WXU,FPRIM,WXXPRM)
      CHARACTER MSG*60
      DIMENSIONFM(NOIM,NOIM),WXX(NOIM,NOIM),BM(NOIM,NOIM),WUU(NOIM,NOIM)
     1,GCSTR(NOIM,NOIM)
      DIMENSION WM1(NOIM,NOIM),WM2(NOIM,NOIM),WM3(NOIM,NOIM),WM4(NOIM,NO
     1IM),WM5(NOIM,NOIM),WM6(NOIM,NOIM),WXU(NOIM,NOIM),
     1 WXXPRM(NOIM,NOIM),FPRIM(NOIM,NOIM)
      DIMENSION COM1(1),COM2(1)
      COMMON /MAIN2/COM2
      COMMON /MAIN1/NOIM,NOIM1,COM1
      COMMON / INOU/ KIN,KOUT,KPUNCH
      COMMON /MAUN5/ MSG
C
C***DETERMINISTIC CONTROLLER GAIN CALCULATION---MODULE # 1
C
C       THIS MODULE COMPUTES THE STEADY STATE DETERMINISTIC CONTROLLER
C       GAIN MATRIX,GCSTAR=(WUUI)(BMT)(KCSSPM).  WUUI IS THE INVERSE OF T
C       CONTROL COST WEIGHTING MATRIX, KCSSPM IS THE STEADY STATE
C       SOLUTION TO  D(KC)/DT=(FMT)(KC)+(KC)(FM)+WXX-(KC)(BM)(WUUI)(BMT)*
C       (KC),  WXX IS THE COST WEIGHTING MATRIX ON THE STATES
C
C       KLEINMAN ROUTINES ARE EXTENSIVELY USED IN THIS MODULE
```

```
C
C
C  TRANSFORM SYSTEM SO THAT WXU NOT=0 CAN STILL BE HANDLED BY
C KLEINMAN ROUTINES.  SEE KWAKERNAAK AND SIVAN'S BOOK, PAGE 322
      CALL PRIMIT(WM1,WUU,ICBM,GCSTR,WXU,IHOLD,BM,WM2,FPRIM,WXX,WXXPRM,       — A
     1 FM)
C NOW HAVE FPRIM, WXXPRM CAN USE RICCATI SOLVER FOR KCSSPM
      MT=1
      CALL TRANS2(IHOLD,ICBM,BM,WM2)                                          — A
C WM2=BMT   ICBM X IHOLD
      CALL EQUATE(WM1,WUU,ICBM,ICBM)
C  GMINV DESTROYS THE CALLING ARRAY
      CALL GMINV(ICBM,ICBM,WM1,WM3,MR,MT)
C WM3=WUUI  ICBM X ICBM-------MR IS AN ERROR INDICATOR
      IF (MR.NE.ICBM) THEN
      PRINT*,'AN ERROR OCCURRED IN INVERTING WUU, MR=',MR,'ICBM=',ICBM
      END IF
      CALL EQUATE(WM1,WM3,ICBM,ICBM)
C WM1=WUUI  SAVE FOR LATER COMPUTATIONS
      CALL MAT1(WM3,WM2,ICBM,ICBM,IHOLD,WM4)                                  — A
C WM4=(WUUI)(BMT)  ICBM  X  IHOLD
C     NOW CALL RICCATI EQUATION SOLVER
      CALL MAT1(BM,WM4,IHOLD,ICBM,IHOLD,WM3)                                  — A
C WM3=BM(WUUI)(BMT)  IHOLD  X  IHOLD
      CALL MRIC(IHOLD,FPRIM,WM3,WXXPRM,WM2,WM6)                               — A
C WM2=KCSSPM   IHOLD X IHOLD
C WM5=FM-BM(WUUI)(BMT)(KCSSPM)---I DONT USE THIS RESULT
      MSG='KCSSPM FOR THE DETERMINISTIC CONTROLLER IS'
      IC=5
      NSAV=NOIM1
      NOIM1=NOIM
      CALL MMATIO(WM2,IHOLD,IHOLD,IC,KIN,KOUT,NOIM,NOIM1)                     — A
C      NOW CALCULATE OPTIMAL GAIN MATRIX GCSTAR.  NOTE  I NEED THE
C      NEGATIVE OF GCSTAR FOR THE CONTROL LAW GENERATION FROM AN LQG
C      CONTROLLER, AND THIS WILL BE THE GCX REQUIRED IN THE PERFORMANCE
C      ANALYSIS ROUTINE
      NOIM1=NSAV
C NOW HAVE KCSSPM , CALCULATE GCSTR=WUUI(BMT(KCSSPM)+WXUT)
C  RECALL WUUI IN WM1
      C1=1.0
      CALL MAT4A(BM,WM2,ICBM,IHOLD,IHOLD,WM4)                                 — A
      CALL TRANS2(IHOLD,ICBM,WXU,WM3)
      CALL MADD1(ICBM,IHOLD,WM4,WM3,WM2,C1)                                   — A
      CALL MAT1(WM1,WM2,ICBM,ICBM,IHOLD,GCSTR)
      NSAV=NOIM1
      NOIM1=NOIM
      IC=5
C GCSTR  ICBM  X  IHOLD
C
      MSG='THE OPTIMAL STEADY STATE FEEDBACK GAIN MATRIX, GCSTR'
      CALL MMATIO(GCSTR,ICBM,IHOLD,IO,KIN,KOUT,NOIM,NOIM1)                    — A
      NOIM1=NSAV
C
      END
* DECK CKFTR
      SUBROUTINE CKFTR(FM,GM,R,HM,NUMDTS,RKFSS,Q,WM1,WM2,
     1WM3,WM4,WM5,WM6,IRFM,IRHM,ICGM,F2,H2,BM,ICBM)
C  CALLING PROGRAM MUST SUPPLY EIGHT WORK SPACE ARRAYS
C
CTHIS ROUTINE CALCULATES  THE KALMAN FILTER GAINS WHEN
C GIVEN THE FM,HM, GM AND R MATRICES AND THE NUMBER OF
C  DETERMINISTIC STATES.   THE CONTROLLER MODEL MUST BE
C  SPECIFIED SUCH THAT ALL THE DETERMINISTIC STATES APPEAR
C  FIRST AND TOGETHER,   THAT IS
CD/DT(X1,X2,...XK,XL,XM,...XN)T=F1     0        B1       .
C                                  (X) +     (U) +  (W)
```

```
C                                      M     F2       B2       G2
C  WHERE X1 THROUGH XK ARE THE DETERMINISTIC STATES AND THE
C  REMAINING STATES ARE STOCHASTIC.  F1 IS K  X  K, AND F2 IS N-K  X
C  N-K,  AND B1,B2,AND G2 ARE PARTITIONED ACCORDINGLY.
C  THIS ROUTINE FIRST STRIPS OFF THE DETERMINSTIC STATES THEN COMPUTES
CAND RETURNS  KALMAN FILTER GAINSFOR THE REMAINING STATES.
C  THE KALMAN FILTER GAINS FOR THE DETERMINSTIC STAES ARE SET TO ZERO
C AND THE KALMAN FILTER GAIN THAT IS RETURNED IS   0
C                                                  RKFSS
C WHERE THE DIMENSION OF THE ZERO VECTOR IS K AND THE RKFSS IS THE
C  STEADY STATE KALMAN FILTER GAIN MATRIX FOR THE N-K STOCHASTIC STATES.
C  THIS AUGMENTED MATRIX IS RETUTNED IN RKFSS
C  ALSO NOTE THAT IN ORDER TTO GENERATE THE KALMAN FILTER, ONLY
C MEASUREMENTS OF STACHASTIC STATES ARE NEEDED SO THE H MATRIX IS
C REDUCED ACCORCDINGLY.
      CHARACTER MSG*60,MSG1*1
      DIMENSION F2(NDIM,NDIM),H2(NDIM,NDIM),FM(NDIM,NDIM),GM(NDIM,NDIM),
     1  R(NDIM,NDIM),HM(NDIM,NDIM),WM1(NDIM,NDIM),Q(NDIM,NDIM),
     1  WM2(NDIM,NDIM),WM3(NDIM,NDIM),WM4(NDIM,NDIM),WM5(NDIM,NDIM)
     1   ,WM6(NDIM,NDIM)
      REAL RKFSS(NDIM,NDIM),BM(NDIM,NDIM)
      DIMENSION COM1(1),COM2(1)
      COMMON /MAIN2/COM2
      COMMON /MAIN1/NDIM,NDIM1,COM1
      COMMON / INOU/ KIN,KOUT,KPUNCH
      COMMON /MAUN5/ MSG
C
C***KALMAN FILTER STEADY STATE GAIN----MODULE # 2
C
C     RKFSS=(PMSS)(HMT)(RI), WHERE PM IS THE STEADY STATE SOLUTION TO TH
C     RICCATI EAUATION D(PM)/DT=FM(PM)+PM(FMT)+GM(Q)(GMT)-
C      PM(HMT)(RI)(HM)(PM)
C
C
C  *******DELETE DETERMINISTIC STATES,  AND ---TRANSPOSE THE F2
C  MATRIX FOR THE RICATTI SOLVER SINCE IT TRANSPOSES THE CALLING ARRAY>
      WRITE(KOUT,*)'IF YOU PLAN TO USE THE DOYLE AND STEIN TECHNIQUE FO
     1R THIS RUN YOU MAY WISH TO MODIFY THE VALUE OF NUMDTS,  THE NUMBER
     1 OF DETERMINISTIC STATES.  DO YOU WANT TO CHANGE NUMDTS?  Y  OR  N
     1>'
      READ(KIN,11)MSG1
      NUMSAV=NUMDTS
      IF (MSG1.EQ.'Y') THEN
      WRITE(KOUT,*)'ENTER THE NEW VALUE OF NUMDTS FOR THIS RUN>'
      READ(KIN,*)NUMDTS
      END IF
      IDS=NUMDTS+1
      DO 2112 I=IDS,IRFM
      II=I-NUMDTS
      DO 2112 J=IDS,IRFM
      JJ=J-NUMDTS
 2112 F2(JJ,II)=FM(I,J)
      IRF2=IRFM-NUMDTS
      DO 2113 I=1,IRHM
      DO 2113 J=IDS,IRFM
      JJ=J-NUMDTS
 2113 H2(I,JJ)=HM(I,J)
C  NOW FORM B2,G2
      DO 2114 I=IDS,IRFM
      II=I-NUMDTS
      DO 2114 J=1,ICGM
 2114 WM1(II,J)=GM(I,J)
C WM1 =G2   IRF2  X  ICGM
      DO 2115 I=IDS,IRFM
      II=I-NUMDTS
      DO 2115 J=1,ICGM
```

```
2113  WM4(II,J)=BM(I,J)
C WM4 =B2  IRF2  X  ICBM
      IRM2=IRMM
      CALL MAT3(IRF2,ICBM,WM1,Q,WM2)
C WM2=GM(Q)(GMT)  IRF2 X IRF2 --USED AS 'Q'IN KLIENMAN RICCATI ROUTINE
      WRITE(KOUT,*)'DO YOU WISH TO MODIFY Q BY THE DOYLE AND STEIN TECHNI
     1QUE,  Y  OR N>'
      READ (KIN,11)MSG1
11     FORMAT(A1)
      IF (MSG1.EQ.'Y') THEN
        CALL DAS1(WM2,WM4,WM1,ICBM,WM3,IRF2)
      END IF
      MT=1
      CALL EQUATE(WM1,R,IRM2,IRM2)
C  GMINV DESTROYS THE CALLING ARRAY
      CALL GMINV(IRM2,IRM2,WM1,WM3,MR,MT)
      IF (MR.NE.IRM2) THEN
      WRITE(KOUT,*)'MR=',MR,'IRM2=',IRM2
      WRITE(KOUT,*)'R-INVERSE IS FOULED UP'
      END IF
C WM3= RI  IRM2 X IRM2
      CALL TRANS2(IRM2,IRF2,H2,WM4)
C WM4= H2T IRF2 X IRM2
      CALL MAT1(WM4,WM3,IRF2,IRM2,IRM2,WM5)
C WM5= H2T(RI)   IRF2 X IRM2
      CALL MAT1(WM5,H2,IRF2,IRM2,IRF2,WM3)
C WM3= H2T(RI)(H2)     IRF2 X IRF2
      CALL MRIC(IRF2,F2,WM3,WM2,WM6,WM4)
C  NOW CALL RICCATI EQUATION SOLVERTO GET PMSS
C WM6=PMSS    IRF2 X IRF2
      CALL MAT1(WM6,WM5,IRF2,IRF2,IRM2,WM1)
C WM1=RKFSS  IRF2  X IRM2
      IO=5
C  FORM RKFSS WITH ZEROS ADDED FOR DETER. STATES.
      PRINT*, 'NUMOTS=',NUMOTS
      IF (NUMOTS.NE.0)THEN
      DO 2119 J=1,IRMM
      DO 2118 I=1,NUMOTS
2118  RKFSS(I,J)=0
      DO 2119 I=IOS,IRFM
      II=I-NUMOTS
2119  RKFSS(I,J)=WM1(II,J)
      ELSE
      CALL EQUATE(RKFSS,WM1,IRFM,IRMM)
      END IF
      MSG='STEADY STATE KALMAN FILTER GAIN MATRIX,RKFSS'
      CALL MMATIO(RKFSS,IRFM,IRMM,IC,KIN,KOUT,NOIM,NOIM)
      NUMOTS=NUMSAV
C
      END
* DECK FRMAUG
      SUBROUTINE FRMAUG(Q,R,FT,BT,GCZ,HT,GCX,BCZ,FC,GCY,BCY,GT,XO,PO,
     1FA,BA,GA,QA,GUA,WM1,WM2,WMA,WMB,WMC,WMD,WME,WMF,
     1MXA,PXA,RA,PXVA,GCZA,IRY,IFLGCZ,IFLGSD)
C  THIS ROUTINE FORMS A SET OF AUGMENTED MATRICES NEEDED BY THE
C   PERFORMANCE ANALYSIS ROUTINES
      DIMENSION Q(NOIM,NOIM),R(NOIM,NOIM),FT(NOIM,NOIM)
     1,BT(NOIM,NOIM),GCZ(NOIM,NOIM),HT(NOIM,NOIM),GCX(NOIM,NOIM),
     1  BCZ(NOIM,NOIM),FC(NOIM,NOIM),BCY(NOIM,NOIM),GT(NOIM,NOIM),
     1  XO(NOIM),PO(NOIM,NOIM),WM1(NOIM,NOIM),WM2(NOIM,NOIM)
      DIMENSION FA(NOIM2,NOIM2),BA(NOIM2,NOIM2),GA(NOIM2,NOIM2),
     1 QA(NOIM2,NOIM2),WMA(NOIM2,NOIM2),WMB(NOIM2,NOIM2),
     1  WMC(NOIM2,NOIM2),WMD(NOIM2,NOIM2),WME(NOIM2,NOIM2),WMF
     1 (NOIM,NOIM),GCY(NOIM,NOIM),PXA(NOIM2,NOIM2),RA(NOIM2,NOIM2),
     1  GCZA(NOIM2,NOIM2),PXVA(NOIM2,NOIM2),GUA(NOIM2,NOIM2)
      REAL MXA(NOIM2)
      INTEGER IFLGSO
      CHARACTER MSG*50
      DIMENSION COM1(1),COM2(1)
      COMMON /MAIN4/NOIM2,NOIM3
      COMMCN /MAIN2/COM2
      COMMCN /MAIN1/NOIM,NOIM1,COM1
      COMMCN / INOU/ KIN,KOUT,KPUNCH
      COMMCN /MAUN5/ MSG
      COMMCN /MAIN6/ ICBT,ICBM,ICFA,ICGA,ICGM,ICGT,ICQA,IRFA,IRFM,IRFT,
     1IRHT,IRQA,IC,LOG,IRHM,NUMDTS
      WRITE(KOUT,*)' ENTER A 5 IF YCU WANT ALL THE AUGMENTED MATRICES PR
     1INTED OUT, A 7 FOR NO MATRICES TO BE PRINTED>'
      READ(KIN,*)IO
      IRFA=IRFT+IRFM
      NSAV1=NOIM
      NSAV2=NOIM1
      NSAV3=NOIM2
      NSAV4=NOIM3
C
C***FORM AUGMENTED MATRICES THAT ARE REQUIRED WHEN FORMING XA
C     WA=(WT  VT)T   IMPLIES THAT QA=   Q  0
C                                       0  R
C
C   FORM QA  IRQA  X  IRQA,  IRQA=IRHT+ICGM
C FOR EQUIVALENT DISCRETE TIME SYSTEMS    IRQA= IRFT+IRHT
      IF(IFLGSO.EQ.0)THEN
      IRQ=ICGT
      ELSE
      IRQ=IRFT
       END IF
      DO 2703 I=1,IRQ
      DO 2703 J=1,IRHT
 2703 WM1(I,J)=0
      DO 2704 I=1,IRHT
      DO 2704 J=1,IRQ
 2704 WM2(I,J)=0
      IFORM=1
      NOIM3=NSAV3
      NOIM2=NSAV1
      CALL AUGMAT(WM2,R,WMO,IFORM,IRHT,IRQ,IRHT,IRHT)
      CALL AUGMAT(Q,WM1,WMC,IFORM,IRQ,IRQ,IRQ,IRHT)
      ICQA=IRQ+IRHT
      IRQA=ICQA
       IFORM=2
      NOIM2=NSAV3
      CALL AUGMAT(WMC,WMO,QA,IFCRM,IRQ,ICQA,IRHT,IRQA)
      MSG=' THE AUGMENTED Q MATRIX IS ,QA'
      CALL MMATIO(QA,IRQA,IRQA,IO,KIN,KCUT,NSAV3,NSAV3)
      IMA=NSAV3-IRFT
C     INITIALIZE PXA MXA AND STORAGE VARIABLES
      DO 5105 IMXA=1,NSAV3
 5105 MXA(IMXA)=0
      DO 5100 IMXA=1,IRFT
 5100 MXA(IMXA)=XC(IMXA)
      MSG='  THE INITIAL XA VECTOR IS'
      CALL MVECIO(MXA,IRFA,IO,KIN,KCUT,NSAV3)
      DO 5101 IPXA=1,IRFT
      DO 5102 JPXA=1,IRFT
 5102 PXA(IPXA,JPXA)=PO(IPXA,JPXA)
      DO 5101 JPX=1,IMA
      JPXA=JPX+IRFT
 5101 PXA(IPXA,JPXA)=0
      DO 5103 IPX=1,IMA
      IPXA=IPX+IRFT
      DO 5103 JPXA=1,IRFA
 5103 PXA(IPXA,JPXA)=0
```

```
      MSG=' THE INTITIAL COVARIANCE MATRIX, PXA IS'
      CALL MMATIO(PXA,IRFA,IRFA,IO,KIN,KOUT,NSAV3,NSAV3)
C***PXVA CALCULATION--- REQUIRED ONLY FIRST TIME THROUGH LOOP
C         PXVA=     BT(GCZ)
C              1/2              R
C                         BCZ
C     TO USE THE KLIENMAN MULTIPLY ROUTINES , THE DECLARED DIMENSION OF
C      ARRAY ARGUMENTS MUST BE THE SAME.  THEREFORE IT IS NECESSARY TO
C      FORM GCZA SUCH THAT GCZA(I,J)=GCZ(I,J) FOR I=1,IRHT,  AND J=1,
C      ICBM,  AND ZERO ELSEWHERE
C     THE SAME REASON REQUIRES CALCULATION OF RA
      IT=NSAV3-IRHT
      JT=NSAV3-ICBM
      DO 6013 IG=1,IRHT
      DO 6014 JG=1,ICBM
6014  GCZA(IG,JG)=GCZ(IG,JG)
      DO 6013 JG=1,JT
      JGA=ICBM+JG
6013  GCZA(IG,JGA)=0
      DO 6015 IG=1,IT
      IGI=IG+IRHT
      DO 6015 JG=1,NSAV3
6015  GCZA(IGI,JG)=0
      IR=NSAV3-IRHT
      DO 6017 IRI=1,IRHT
      DO 6017 JRI=1,IRHT
6017  RA(IRI,JRI)=R(IRI,JRI)
      DO 6016 JRI=1,IR
      JRJ=JRI+IRHT
6016  RA(IRI,JRJ)=0
      DO 6018 IRI=1,IR
      IRII=IRI+IRHT
      DO 6018 JRI=1,NSAV3
6018  RA(IRII,JRI)=0
C       RA = R IN UPPER LEFT PARTITION,ZERO ELSEWHERE
      IF((IFLGCZ.EQ.0).AND.(IFLGSD.EQ.1)) THEN
C CALCULATE PXVA ONLY FOR GCZ NOT  EQUAL TO ZERO MATRIX AND NOT FOR S-D
C  RECALL THAT (BT(GCZ)  BCZ)T  IS THE RIGHR PARTITION OF GA
      DO 6000 IPXA=1,IRHT
      IPA=IPXA+ICGT
      DO 6000 JPXA=1,IRFA
6000  PXVA(JPXA,IPXA)=GA(JPXA,IPA)
      C1=.5
      NDIM=NSAV3
      NDIM1=NSAV3+1
      CALL MSCALE(WMF,PXVA,IRFA,IRHT,C1)
      CALL MAT1(WMF,RA,IRFA,IRHT,IRHT,PXVA)
C PVXA=PXAVT    IRFA   X   IRHT
      MSG=' CROSS COVARIANCE, PXVA IS'
      CALL MMATIO(PXVA,IRFA,IRHT,IO,KIN,KOUT,NSAV3,NSAV3)
      END IF
C
C
C          FA= FA11  FA12  =FT+BT(GCZ)(HT)        BT(GCX)
C             =FA21  FA22  =BCZ(HT)               FC
C
C
C  WHERE GCZ IS THE GAIN MATRIX THAT ACTS DIRECTLY ON THE MEASUREMENT
C       VECTOR,  AND GCX IS THE GAIN MATRIX THATS ACTS ON THE CONTROLLER
C       STATE ESTIMATES.*****THESE MUST BE SUPPLIED BY THE GAIN MATRIX
C       ROUTINE---------------C
C
C***FORM FA
      NDIM1=NSAV1+1
      NDIM=NSAV1
C
```

```
      CALL MAT1(BT,GCZ,IRFT,ICBT,IRHT,WM1)
C WM1=BT(GCZ) IRFT X IRHT
      CALL MAT1(WM1,HT,IRFT,IRHT,IRFT,WM2)
C WM2= BT(GCZ)HT  IRFT X IRFT
      C1=1.0
      CALL MADD1(IRFT,IRFT,FT,WM2,WM1,C1)
C WM1= FA11  IRFT X IRFT
      IF (ICBM.NE.ICBT)THEN
      WRITE(KOUT,*)'ICBM=',ICBM,' ICBT=',ICBT
      WRITE(KOUT,*)'BT ANC BM ARE NCT THE SAME SIZE- WILL CAUSE ERRORS'
      END IF
      CALL MAT1(BT,GCX,IRFT.ICBT,IRFM.WM2)
C WM2=FA12    IRFT XIRFM
      IFORM=1
      NOIM2=NSAV1
      CALL AUGMAT(WM1,WM2,WMA,IFORM,IRFT,IRFT,IRFT.IRFM)
C WMA= (FA11  FA12)    IRFT X IRFT+IRFM
      CALL MAT1(BCZ,HT,IRFM,IRHT,IRFT,WM1)
C WM1= FA21   IRFM  X  IRFT
      CALL AUGMAT(WM1,FC,WMB,IFORM,IRFM,IRFT,IRFM,IRFM)
C WMB= (FA21  FA22)    IRFM  X IRFM+IRFT
      NOIM2=NSAV3
      IFORM=2
      IRFA=IRFT+IRFM
      ICFA=IRFA
      CALL AUGMAT(WMA,WMB,FA,IFORM,IRFT,IRFA,IRFM,ICFA)
      MSG='THE AUGMENTED F MATRIX FA IS'
      CALL MMATIO(FA,IRFA,IRFA,IO.KIN.KOUT,NSAV3,NSAV3)
C FA   IRFA  X  IRFA
C
C***FORM BA - - - FOR REGULATOR CASE , NOT REQUIRED Y=0
       CALL MAT1(BT,GCY,IRFT,ICBT,IRY,WM1)
C  WM1= BT(GCY)  IRFT   X  IRY
      IFORM=2
      NOIM2=NSAV1
      CALL AUGMAT(WM1,BCY,BA,IFORM,IRFT,IRY,IRFM,IRY)
      MSG=' THE AUGMENTED B MATRIX BA,IS'
      CALL MMATIO(BA,IRFA,IRY,IO,KIN,KOUT,NSAV3,NSAV3)
C   BA IRFA  X  IRY

C
C***FORM GA
C
C   GA=  GT           BT(GCZ)
C     =  0             BCZ
C
      CALL MAT1(BT,GCZ,IRFT,ICBM,IRHT,WM1)
C WM1= BT(GCZ)  IRFT  X  IRHT
      IFORM=1
      CALL AUGMAT(GT,WM1,WMC,IFORM,IRFT,IRQ,IRFT,IRHT)
C  RECALL IRQ=ICGT FOR CONTINUOUS SYS,=IRFT FOR S-D SYS
C WMC= (GT   BT(GCZ))   IRFT  X  IRHT+IRQ
      DO 3001 IR=1,IRFM
      DO 3001 IC=1,IRQ
3001  WM1(IR,IC)=0
      CALL AUGMAT(WM1,BCZ,WMO,IFORM,IRFM,IRQ,IRFM,IRHT)
C WMO=(0    BCZ)     IRFM  X IRQ+IRHT
      ICGA=IRQ+IRHT
      IFORM=2
      NOIM2=NSAV3
      CALL AUGMAT(WMC,WMO,GA,IFORM,IRFT,ICGA,IRFM,ICGA)
      MSG=' THE AUGMENTED  G MATRIX GA , IS'
      CALL MMATIO(GA,IRFA,ICGA,IO,KIN,KCUT,NSAV3,NSAV3)
C GA   IRFA  X  ICGA
C
C   GUA=GCZ(HT)   GCX)
```

```
      NOIM=NSAV1
      NOIM1=NSAV1+1
       CALL MAT1(GCZ,HT,ICBT,IRHT,IRFT,WM2)
C WM2= GCZ(HT)      ICBT X IRFT
      IFORM=1
      NOIM2=NSAV1
      NOIM3=NSAV3
      CALL AUGMAT(WM2,GCX,GUA,IFORM,ICBT,IRFT,ICBM,IRFM)
      MSG='THE AUGMENTED MATRIX GUA IS'
      CALL MMATIO(GUA,ICBT,IRFA,IC,KIN,KOUT,NSAV3,NSAV3)
C  GUA  ICBT  X  IRFA
C******AUGMENTED SYSTEM MATRICES NOW AVAILBLE FOR COMPUTATION
C
      NOIM=NSAV1
      NOIM1=NSAV2
      NOIM2=NSAV3
      NOIM3=NSAV4
      END
* DECK PERFAL
      SUBROUTINE PERFAL(IRY,IFLGCZ,MXA,GCY,GUA,PXA,PXVA,IFLGSO,
     1RA,GCZA,YO,EAT,INTGA,WME,WMF,PUU,WM1,MUOUT,MXTOUT,PUOUT,PXTOUT,
     1MXAMIN,MXAMAX,PXTMIN,PXTMAX,PUMIN,MUMAX,PUMIN,PUMAX,MU,INTBA,
     1WV3,WV4)
      CHARACTER MSG*60
      REAL WM1(NOIM,NOIM),EAT
     1(NOIM2,NOIM2),INTGA(NOIM2,NOIM2),WME(NOIM2,NOIM2),
     1WMF(NOIM2,NOIM2),WV3(NOIM2),WV4(NOIM2),
     1    MXA(NOIM2),PXA(NOIM2,NOIM2),PXVA(NOIM2,
     1NOIM2),MUOUT(NOIM),MXTOUT(NOIM),PXTOUT(NOIM),PUOUT(NOIM),
     1    YO(NOIM3),MXAMIN(NOIM),MXAMAX(NOIM),MUMIN(NOIM),MUMAX(NOIM),
     1    PXTMIN(NOIM),PXTMAX(NOIM),PUMIN(NOIM),PUMAX(NOIM),GCZA
     1    (NOIM2,NOIM2),RA(NOIM2,NOIM2),GUA(NOIM2,NOIM2),GCY(NOIM,
     1    NOIM),INTBA(NOIM2,NOIM2)
      INTEGER IFLGCZ
      DIMENSION COM1(1),COM2(1)
      REAL MU(NOIM),PUU(NOIM2,NOIM2)
      COMMON /RNTIM/ RNTIME,DELTIM
      COMMON /MAIN4/NOIM2,NOIM3
      COMMON /MAIN2/COM2
      COMMON /MAIN1/NOIM,NOIM1,COM1
      COMMON / INOU/ KIN,KOUT,KPUNCH
      COMMON /MAUNS/ MSG
      COMMON/MAIN6/ICBT,ICBM,ICFA,ICGA,ICGM,ICGT,ICCA,IRFA,IRFM,IRFT,
     1IRHT,IRQA,IO,LOG,IRFM,NUMOTS
      NSAV1=NOIM
      NSAV2=NOIM1
      NSAV3=NOIM2
      NSAV4=NOIM3
C
C****PERFORMANCE ANALYSIS ROUTINE
C     THIS IS A CONTINUOUS TIME MEASUREMENT PERFORMANCE ANALYSIS
C     ROUTINE FOR EVALUATING CONTINUCUS TIME CONTROL SYSTEMS DRIVEN BY
C      WHITE GAUSSIAN NOISE.   IT COMPUTES THE MEAN AND COVARIANCE OF THE
C      OF THE TRUTH MODEL STATES ,THE CONTROLLER STATES,AND THE CONTROLS
C      GENERATED.   A SET OF AUGMENTED MATRICES IS USED TO DO THE
C       CALCULATIONS---Y=(XT  USTAR)T,    XA=(XT  XM)T .. THE PERFORMANCE
C       ANALYSIS ROUTINE IS DEVELOPED IN A MASTERS THESES FOR AIR FORCE
C       INSTITUTE OF TECHNOLOGY BY ERIC LLOYD,  TITLE 'ROBUST CONTROL
C       SYSTEM DESIGN'
C
C
C***MXA,PXA CALCULATION--- THE MEAN  AND COVARIANCE OF THE XA VECTOR
C         FOUND USING SOLUTION FORMS OF THE PROPAGATION EQUATONS
C       KLIENMAN ROUTINES ARE USED TO PROVIDE THE SOLUTIONS
C  IN THE FOLLOWING TWO EQS, THE FIRST OCCURRANCE OF PXA OR MXA
C  IS THE VALUE AT TIME T+DELTIM, THE SECOND ----AT TIME T
```

```
C          PXA=EAT(PXA)EATT+INTGA
C              MXA=EAT(MXA)+INTBA
C     SEE DEFINITIONS BELOW FOR EAT ,INTGA,  INTBA
C
C        NOTE SINCE THIS PROGRAM CONSIDERS ONLY THE REGULATOR CASE,
C         Y- THE DESIRED INPUT-  IS ASSUMED = ZERO
C
C
C        MXAO=(E(XO)  0)T  =(XO   0)T    E=THE EXPECTED VALUE OPERATOR
C        PXAO=   PO  0
C                 )   0
C
C   EAT=   EXP(FA*TIME)
C   INTBA=  INTEG(EAT)BA    FOR CONTINUOUS TIME SYSTEMS
C        =  BAD FOR DISCRETE SYSTEMS
C   INTGA=   INT(EAT(GA)QA(GA T)EATT)  FOR CONT TIME SYSTEMS
C       =  GDA(CDA)GDAT    FOR DISCRETE TIME SYSTEMS
C
.      LCOUNT=0
       IRN=NINT(RNTIME/DELTIM)
       WRITE(KOUT,*)'ENTER A  0  IF YOU WANT NO PRINTS OF PXA, PUU, MXA,
     1 AND MU MATRICES DURING THE PERFORMANCE ANALYSIS,  ELSE ENTER THE
     1 NUMBER OF TIME INCREMENTS BETWEEN PRINTS( THERE ARE ',IRN,' TOTAL
     1 TIME INCREMENTS IN THIS RUN)>'
       READ(KIN,*)IPCNTL
       IF (IPCNTL.EQ.0) THEN
       IC=0
       ELSE
       IO=5
       END IF
       DO  5003 IN=1,NSAV1
       PUU(IN,IN)=0
       MU(IN)=0
       MXAMIN(IN)=0
       MXAMAX(IN)=0
       PXTMIN(IN)=0
       PXTMAX(IN)=0
       MUMIN(IN)=0
       MUMAX(IN)=0
       PUMAX(IN)=0
       PUMIN(IN)=0
 5003  CONTINUE
       WRITE(KOUT,*)'ENTER THE NUMBER OF SAMPLE PERIODS DESIRED BETWEEN
     1 PLOT POINTS(MAX 1000 PLOT POINTS) THERE ARE ',IRN,' SAMPLE PERIOD
     1S REQUESTED FOR THIS RUN>'
       READ(KIN,*)IPLTPS
       DELPLT=DELTIM*IPLTPS
       DO 5000 ITLMP=1,IRN
C
       IF(IC.EQ.5) THEN
       JJ=ITLMP-1
       IPNT=MOD(JJ,IPCNTL)
       IF((IPNT.EQ.0).OR.(ITLMP.EQ.IRN)) THEN
       TIME=JJ*DELTIM
       WRITE(KOUT,*)'TIME=  ',TIME
       MSG='     PXA'
       CALL MMATIO(PXA,3,3,IC,KIN,KOUT,NSAV3,NSAV3)                      M
       MSG='     PUU'
       CALL MMATIO(PUU,IC3M,IC8M,IC,KIN,KOUT,NSAV3,NSAV3)
       MSG='MXA'
       CALL MVECIO(MXA,3,IO,KIN,KOUT,NSAV3)                              M
       MSG='MU'
       CALL MVECIO(MU,IC8M,IO,KIN,KOUT,NSAV1)
       END IF
       END IF
C      NOW WANT TO STORE FOR PLOTTING, MXT,PXY,MU,PUU
```

```
      DO 123 IWR=1,7
      MXTOUT(IWR)=MXA(IWR)
      PXTOUT(IWR)=SQRT(PXA(IWR,IWR))
123   CONTINUE
      DO 124 IWR=1,ICBM
      MUOUT(IWR)=MU(IWR)
      PUOUT(IWR)=SQRT(PUU(IWR,IWR))
124   CONTINUE
C*********
C****** NO CROSS CORRELATION TERMS ARE PLOTTED
C
      NDIM6=NSAV1
      IPTCTL=MOD(JJ,IPLTPS)
      IF (IPTCTL.EQ.0) THEN
      CALL STORED(LOG,RATIME,DELPLT,LCOUNT,7,7,ICBM,ICBM,
     1 MXTOUT,PXTOUT,MUOUT,PUOUT,NDIM6)
      LCOUNT=LCOUNT+1
      END IF
C  NOTE THAT YD IS RESTRICTED BY VALUE OF LCOUNT TO BE CONSTANT
C  BETWEEN PLOT POINTS
      IF (LCOUNT.GT.1000)THEN
C RESET LCOUNT
      LCOUNT=1000
      END IF
C***UPDATE PXA,MXA
C
C
      NDIM=NSAV3
      NDIM1=NSAV3+1
      CALL MAT3(IRFA,IRFA,EAT,PXA,WME)
      C1=1.0
      CALL MADD1(IRFA,IRFA,WME,INTGA,PXA,C1)
C***PXA AT NEW TIME NOW AVAILABLE
C  MXA= EAT(MXA0)+INTEG(EAT(BA))(YD)
      DO 1814 IK=1,IRFA
1814  WV3(IK)=0
      DO 1815 IK=1,IRFA
      DO 1815 IJ=1,IRFA
1815  WV3(IK)=WV3(IK)+EAT(IK,IJ)*MXA(IJ)
      DO 1812 IWR=1,IRFA
1812  WV4(IWR)=INTBA(IWR,1)*YD(LCOUNT)
      DO 1813 IJ=1,IRFA
1813  MXA(IJ)=WV3(IJ)+WV4(IJ)
C***MXA AT NEW TIME NOW AVAILABLE
C***MU,PUU CALCULATION FOR ZERO MEAN MEASUREMENT NOISE
C  MU=GUA(MXA)+GCZ(MVT)+GCY(YD)....MVT ,THE MEAN OF NOISE V ASSUMED =0
      X1=YD(LCOUNT)
      NDIM=NSAV1
      NDIM1=NSAV1+1
      CALL MSCALE(WM1,GCY,ICBM,IRY,X1)
C WM1= GCY(YD       ICBM  X  1
      NDIM=NSAV3
      NDIM1=NSAV3+1
      DO 1817 IJ=1,ICBM
1817  WV3(IJ)=0
      DO 1816 IJ=1,ICBM
      DO 1816 IK=1,IRFA
1816  WV3(IJ)=WV3(IJ)+GUA(IJ,IK)*MXA(IK)
C WV3=GUA(MXA)    ICBM   X   1
C   ADDED TO WM1 ABOVE TO GET MU
      DO 329 I=1,ICBM
329   MU(I)=WV3(I)+WM1(I,1)
C MU    IRFA  X 1  ----NOW AVAILABLE------
C***PUU CALCULATION,PUU=GUA(PXA)GUAT+GUA(PXVA)GCZT+GCZ(PXVA)GUAT+
C          GCZ(R)GCZT
      CALL MAT3(ICBM,IRFA,GUA,PXA,PUU)
```

```
C PUU=GUA(PXA)GUAT   ICBM  X  ICBM
C******IFLGCZ*****
      IF (IFLGCZ.EQ.0) THEN
C        SINCE GCZ NOT EQUAL TO ZERO CALCULATE OTHER TERMS OF PUU
C******IFLGCZ******
      IF (IFLGSO.EQ.0) THEN
C DONT DO THIS FOR S-O CASE
      CALL MAT1(GUA,PXVA,ICBM,IRFA,IRHT,WMF)
      CALL MAT4(WMF,GCZA,ICBM,IRHT,ICBM,WME)
      C1=1.0
      CALL MADD1(ICBM,ICBM,PUU,WME,WMF,C1)
C WMF=GUA(PXA)GUAT+GUA(PXVA)GCZT
      CALL MAT4(GCZA,PXVA,ICBM,IRHT,IRFA,PUU)
      CALL MAT4(PUU,GUA,ICBM,IRFA,ICBM,WME)
      CALL MADD1(ICBM,ICBM,WME,WMF,PUU,C1)
C PUU=GUA(PXA)GUAT+GUA(PXVA)GCZT+GCZ(PVXA)GUAT
      END IF
      CALL MAT3(ICBM,IRHT,GCZA,RA,WME)
      CALL MADD1(ICBM,ICBM,WME,PUU,WMF,C1)
      CALL EQUATE(PUU,WMF,ICBM,ICBM)
      END IF
C*** PUU NOW AVAILABLE  ICBM  X  ICBM
C
C
C
5030  CONTINUE
      DO 120 IWR=1,7
      MXTOUT(IWR)=MXA(IWR)
120   PXTOUT(IWR)=SQRT(PXA(IWR,IWR))
      DO 121 IWR=1,ICBM
      MUOUT(IWR)=MU(IWR)
121   PUOUT(IWR)=SQRT(PUU(IWR,IWR))                                   M
      CALL STORED(LOG,RNTIME,DELTIP,LCOUNT,7,7,ICBM,ICBM,
     1 MXTOUT,PXTOUT,MUOUT,PUOUT,NDIM6)
C      CALL TO STORED  WITH LCOUNT < 0 INDICATE THIS DATA RUN  COMPLETE
      LCOUNT=-12
      CALL STORED(LOG,RNTIME,DELTIP,LCOUNT,7,7,ICBM,ICBM,
     1    MXTOUT,PXTOUT,MUOUT,PUOUT,NDIM6)
      NDIM=NSAV1
      NDIM1=NSAV2
      NDIM2=NSAV3
      NDIM3=NSAV4
C   RESET IO TO SOME NONZERO  VALUE TO AVOID TERMINATING THE PROGRAM
C   WHEN RETURNING  TO MAIN ROUTINE.  LOGRP
      IO=25
      END
* DECK MYPLOT
      SUBROUTINE MYPLOT
      END
* DECK AUGMAT
      SUBROUTINE AUGMAT(A1,A2,A3,IFORM,IRA1,ICA1,IRA2,ICA2)
C
C*****NDIM2,NDIM3 MUST BE SET IN THEM CALLING PROGRAM BEFORE USING
C*****NDIM2,NDIM3 MUST BE SET IN THE CALLING PROGRAM BEFORE USING
C*****    THIS SUBROUTINE.   THEY MUST BE DECLARED IN A COMMON BLOCK
C        LABELED --MAIN4---
* DECK MVECIO
C     THIS SUBROUTINE FORMS AUGMENTED MATRICES OF THE FORM
C         IFORM=1          A3=(A1  A2)
C         IFORM=2          A3=(A1  A2)T
C
C
C     IRA1,IRA2,ARE ROW DIMENSIONS,ICA1,ICA2,ARE COLUMN DIMENSIONS
      DIMENSION A1(NDIM2,NDIM2),A2(NDIM2,NDIM2),A3(NDIM3,NDIM3)
      COMMON /MAIN4/NDIM2,NDIM3
      IF (IFORM.EQ.1) THEN
```

```
C  FORM THE AUGMENTED MATRIX A3=(A1  A2)
      DO 12 II=1,IRA1
      DO 11 III=1,ICA1
11    A3(II,III)=A1(II,III)
      DO 12 IV=1,ICA2
      IVI=IV+ICA1
12    A3(II,IVI)=A2(II,IV)
      RETURN
      END IF
C FORM AUGMENTED MATRIX A3=(A1  A2)T
      IFAJ=IRA1+IRA2
      ICA3=ICA1
      DO 14 II=1,ICA3
      DO 13 III=1,IRA1
13    A3(III,II)=A1(III,II)
      DO 14 IV=1,IRA2
      IVI=IV+IRA1
14    A3(IVI,II)=A2(IV,II)
      RETURN
      END
      SUBROUTINE MVECIO(A,NUMEL,IC,KIN,KOUT,NDIM)
C   THIS SUBROUTINE READS PRINTS ENTIRE (PORTIONS OF) THE VECTOR
C   A, DEPENDING ON THE VALUE OF ---IO---.   IO=1---READ ONLY
C   IO=2---READ AND PRINT,   IO=3 READ SELECTED VALUE.  IO=4
C   READ AND PRINT SELECTED VALUES
C    IO=5---PRINT ONLY
C  TO USE IO=3 OR 4  THE CASLLING PFOGRAM MUST INITIALZE THE VEC.
C   ******THIS ROUTINE SETS IO=0---- WHEN NO DATA IN INPUT FILE
C
CREAD IS FROM UNIT SPECIFIED BY CALLING PROGRAM IN KIN,WRITE IS TC
C  KOUT.NDIM IS THE DECLARED DIMENSICN OF A IN THE CALLING
C    PROG AM
      CHARACTER MSG*50
      DIMENSICN A(NDIM)
      COMMCN /MAUNS/ MSG
      IF ((IO.EQ.1).OR.(IO.EQ.2)) THEN
C  READ ENTIRE VECTOR
      WRITE(KOUT,*)'ENTER ',NUMEL,'ELEMENTS>'
      READ(KIN,*,END=29)(A(I),I=1,NUMEL)
      END IF
      IF (IO.EQ.1) THEN
      RETURN
      END IF                                                         M
      IF ((IO.EQ.3).OR.(IC.EQ.4).OR.(IO.EQ.6)) THEN
C   READ ONLY SELECTED ELEMENTS.  THE FIRST NUMBER ON EACH CARD
C IS THE SUBSCRIPT, THE SECONC IS THE DATA ENTRY
C********NCTE ONLY ONE DATA ENTRY PER CARD
C******** FIRST CARD MUST CONTAIN THE TOTAL NUMBERR CF ENTRIES TO BE
C    READ
      IF(IC.NE.6) GO TO 4
      DO 3 I=1,NDIM
 3    A(I)=0.
4     CONTINUE
      WRITE(KOUT,*)'ENTER THE ELEMENT NUMBER ,THEN ITS VALUE>'
5     READ(KIN,*,END=29)I,ENTRY
      IF(I.LE.0) GO TO 20                                            M
      IF(I.GT.NDIM) GO TO 5
      A(I)=ENTRY
      IF (IO.EQ.4)THEN
      WRITE(KOUT,33)' '
      WRITE(KOUT,*)MSG
      WRITE(KOUT,*)'ELEMENT NUMBER , ENTRY>'
      WRITE(KOUT,11)I,A(I)
      END IF
      GO TO 5                                                        M
   20 CONTINUE
```

```
      RETURN
      END IF
      IF ((IO.EQ.2).OR.(IO.EQ.5)) THEN
C   TO GET HERE IO=2 OR 5 SO PRINT OUT ENTIRE VECTOR
      WRITE(KOUT,33)' '
      WRITE(KOUT,*)MSG
      WRITE(KOUT,*)'  THE VECTOR HAS ',NUMEL,'  ELEMENTS'
      WRITE(KOUT,22)(A(I),I=1,NUMEL)
      RETURN
      END IF
      RETURN
   29 PRINT*,'END OF DAT REACHED DURING INPUT IN MVECIO'
      IO=0
C   ******THIS ROUTINE SETS IO=0---- WHEN NO DATA IN INPUT FILE
33    FORMAT(A10,/)
   11 FORMAT(I4,15X,E12.6)
   22 FORMAT(10(10(1X,E12.6),:,/))
      RETURN
      END
* DECK MMATIO
      SUBROUTINE MMATIO(A,IR,IO,IC,KIN,KOUT,NDIM,NDIM1)
      CHARACTER MSG*60
      DIMENSION A(NDIM,NDIM1)
      COMMON /MAUN5/ MSG
C THIS SUBROUTINE READS AND/OR PRINTS THE MATRIX A DEPENDING ON THE
C VALUE OF IO.  IT READS FROM UNIT SPECIFIED BY KIN AND WRITES TO UNIT
C  KOUT.  IO=1--READ ENIRE ARRAY      IO=2--READ AND PRINT ENTIRE
C
C   ARRAY.   IO==3---READ SELECTED ELEMENTS OF A   IO=4---READ AND
C  PRINT SELECTED ELEMENTS OF A       IO=5  ---PRINT ENTIRE ARRAY
C   NDIM,NDIM1 ARE THE DIMENSIONS OF A IN THE CALLING PROGRAM
C
C   ******NOTE IF IO=3 OR 4  THE CALLING PROGRAM MUST INITIALIZE
C         THE ENTIRE ARRAY BEFORE CALL
C
C   *******THIS ROUTINE SETS IO =0 ---- WHEN THE INPUT FILE IS EMPTY
C
      IF ((IO.EQ.1).OR.(IO.EQ.2)) THEN
C   READ ENTIRE ARRAY IN FREE FORMAT,ROW MAJOR ORDER
      WRITE(KOUT,*)'ENTER ',(IR*IC),' ARRAY ELEMENTS IN ROW MAJ ORDER>'
      READ(KIN,*,END=29)((A(I,J),J=1,IC),I=1,IR)
      END IF
      IF (IO.EQ.1) THEN
      RETURN
      END IF
      IF ((IO.EQ.3).OR.(IC.EQ.4).OR.(IO.EQ.6)) THEN                    M
C   READ IN SELECTED ELEMENTS OF A
C   THE FIRST CARD IN THE INPUT STREAM MUST CONTAIN THE TOTAL
C    NUMBER OF ELEMENTS TO BE READ IN.  ONLY ONE ENTRY PER CARD.
C    THE FIRST ITEM ON EACH CARD IS THE ROW,  THESECOND IS THE COL THE
C     LAST ON EACH CARD IS THE DATA FOR THAT LOCATION
C      FREE FORMAT IS USED
      IF(IC.NE.6) GO TO 4
      DO 45 M=1,IR
      DO 45 N=1,IC
45    A(M,N)=0.0
4     CONTINUE                                                          M
      WRITE(KOUT,*)'ENTER THE ROW,AND COLUMN FOLLOWED BY ITS VALUE>'
5     READ(KIN,*,END=29)II,J,ENTRY
      IF((II.LE.0).OR.(J.LE.0)) GO TO 29
      IF((II.GT.IR).OR.(J.GT.IC)) GO TO 5
      A(II,J)=ENTRY
      IF (IO.EQ.4) THEN
      WRITE(KOUT,33)' '
      WRITE(KOUT,*)MSG
      WRITE(KOUT,*)'(',II,',',J,')=',A(II,J)
      END IF
      GO TO 5
   20 CONTINUE
      RETURN
      END IF
      IF ((IO.EQ.2).OR.(IC.EQ.5)) THEN
C   IO = 2 OR 5 IF HERE SC PRINT ENTIRE ARRAY
      WRITE(KOUT,33)' '
      WRITE(KOUT,*)MSG
      WRITE(KOUT,*)'  MATRIX  SIZE IS ',IR,'  X ',IC
      DO 49 I=1,IR
   49 WRITE(KOUT,48)(A(I,J),J=1,IC)
33    FORMAT(A10,/)
   48 FORMAT(5(10(1X,E12.6),/,/))
      END IF
      RETURN
   29 PRINT *,'END OF DATA REACHED DURING INPUT'
C  *******THIS ROUTINE SETS IO =0 ---- WHEN THE INPUT FILE IS EMPTY
      IO=0
      RETURN
      END
* DECK CLOGRS
      SUBROUTINE CLOGRS(GCSTR,FM,BM,RKFSS,HM,GCX,GCY,GCZ,
     1BCY,BCZ,FC,YD,RM,QM,FT,BT,GT,QT,RT,HT,IRY,IFLGCZ,WM1
     1 ,WM2,WM3,PO,GM,WM4,WM5,WM6,
     1  WV1,WV2,WUU,WXX,XO,WXU,WM7,WM8)
C  THIS ROUTINE PERFORMS SET UP FCR USING THE CONTINUOUS TIME
C PERFORMANCE ANALYSIS  FOR ANLOG REGULATOR
      DIMENSIONGCSTR(NDIM,NDIM),FM(NDIM,NDIM),BM(NDIM,NDIM),PO(NDIM,
     1NDIM),HM(NDIM,NDIM),GCX(NDIM,NDIM),GCY(NDIM,NDIM),GCZ(NDIM,NDIM),
     1BCY(NDIM,NDIM),BCZ(NDIM,NDIM),FC(NDIM,NDIM),YD(NDIM3),WM1(NDIM,
     1NDIM),WM2(NDIM,NDIM),WM3(NDIM,NDIM),FT(NDIM,NDIM),
     1 BT(NDIM,NDIM),GT(NDIM,NDIM),HT(NDIM,NDIM),RM(NDIM,NDIM),
     1QT(NDIM,NDIM),RT(NDIM,NDIM),QM(NDIM,NDIM),WV1(NDIM),WV2(NDIM
     1),XO(NDIM),GM(NDIM,NDIM),WUU(NDIM,NDIM),
     1 WXX(NDIM,NDIM),WM4(NDIM,NDIM),WM5(NDIM,NDIM),
     1 WM6(NDIM,NDIM),WXU(NDIM,NDIM),WM7(NDIM,NDIM),WM8(NDIM,NDIM)
      DIMENSION COM1(1),COM2(1)
      CHARACTER MSG*60
      INTEGER IFLGCZ
      REAL RKFSS(NDIM,NDIM)
       COMMON /MAIN4/NDIM2,NDIM3
      COMMCN /MAUN5/ MSG
      COMMCN /MAIN1/ NDIM,NDIM1,COM1
      COMMCN /MAIN2/  COM2
      COMMON /RNTIM/RNTIME,DELTIM
      COMMCN /INOU/ KIN,KOUT,KPUNCH
      COMMCN /MAIN6/ICBT,ICBM,ICFA,ICGA,ICGM,ICGT,ICGA,IRFA,IRFM,IRFT,
     1  IRHT,IRQA,IO,LOG,IRHM,NUMDTS
      WRITE(KOUT,*)'DO YOU WANT TC CALCULATE EIGENVALUES OF THE TRUTH MO
     1DEL AND CONTROLLER MODEL  F  MATRICES,  Y  OR  N>'
      READ(KIN,23,END=2933) MSG
23    FORMAT(A1)
      IF (MSG.EQ.'Y') THEN
      WRITE(KOUT,*)' '
      WRITE(KOUT,*)'THE EIGENVALUES OF THE TRUTH MODEL F MATRIX'
       CALL MEIGN(FT,WV1,WV2,IRFT,WM1)
      WRITE(KOUT,*)'THE EIGENVALUES OF THE CONT. MODEL F MATRIX'
      CALL MEIGN(FM,WV1,WV2,IRFM,WM1)
      END IF
      WRITE(KOUT,*)'COLORED INPUT NOISE (Y OR N) >'
      READ(KIN,23)MSG
      IMOLD=IRFM
      IF(MSG.EQ.'N')GO TO 2900
      CALL CNOISE(FM,GM,HM,BM,QM,IRFM,IRHM,ICGM,ICBM,WM1,WM2,WM3)
2900  CALL CKFTM(FM,GM,BM,HM,NUMDTS,RKFSS,QM,WM1,WM2,WM3,WM4,WM5,WM6,
     1  IRFM,IRHM,ICBM,GCY,GCZ,BM,ICBM)
C   GCZ,GCY,  IN CALL TO CKFTR ARE USED AS DUMMY ARRAYS FOR H2 ,F2
        CALL COTCON(FM,BM,WXX,WUU,GCSTR,IHOLD,ICBM,WM1,WM2,WM3,WM4,WM5,
      1WM6,WAU,WM7,WM8)
        IF(IHOLD.EQ.IRFM) GC TO 35
        ICH=IHOLD+1
        DC 25 I=1,ICBM
        DO 25 J=ICH,IRFM
25      GCSTR(I,J)=0.
35      CONTINUE
        WRITE(KOUT,*)'ENTER THE TOTAL RUN TIME AND THE TIME INCREMENT>'
        READ(KIN,*,END=2933)RNTIME,DELTIM
        C1=-1
        CALL IDNT(IRFM,WM1,C1)
C WM1=-I  IRFM X IRFM
        CALL MAT1(GCSTR,WM1,ICBM,IRFM,IRFM,GCX)
        IO=5
        MSG='GCX FOLLOWS,  GCY,GCZ SET=0'
        CALL MMATIO(GCX,ICBM,IRFM,IC,KIN,KOUT,NDIM,NDIM)
C  NOTE****** OTHER GAIN MATRICES,GCZ,AND GCY SHOULD BE CALCULATED IN
C          THIS MODULE FOR USE IN THE PERFORMANCE ANALYSIS ROUTINE.
        DO 2902 III=1,ICBM
        DO 2902 IIII=1,IRHM
2902    GCZ(III,IIII)=0
        IFLGCZ=1
C   IFLGCZ=1 INDICATES GCZ IS SET TO ZERO--PREF ANALYSIS ROUTINE USES IF
C
        DO 2903 I=1,1000
2903    YD(I)=0
        IRY=1
        DO 1723 I=1,ICBM
        DO 1723 J=1,IRY
1723    GCY(I,J)=0
C  FORM BCY
        DO 1702 I=1,IRFM
        DO 1702 J=1,IRY
1702    BCY(I,J)=0
C  YD IS ALLOWED TO ONLY BE A SCALAR AT THIS TIME
C    FORMBCZ
        C1=1.0
        CALL EQUATE(BCZ,RKFSS,IRFM,IRHM)
        MSG='BCZ FOLLOWS,BCY=0'
        CALL MMATIO(BCZ,IRFM,IRHM,IC,KIN,KOUT,NDIM,NDIM)
C    FORM FC
        CALL MAT1(BM,GCX,IRFM,ICBM,IRFM,WM1)
        CALL MADD1(IRFM,IRFM,FM,WM1,WM2,C1)
        WRITE(KOUT,*)' '
        WRITE(KOUT,*)'DO YOU WANT TO CALCULATE THE EIGENVALUES OF THE CONT
      1INUOUS-TIME LQ CONTROLLER?    Y   OR N>'
        READ(KIN,23)MSG
        IF (MSG.EQ.'Y') THEN
          WRITE (KOUT,*)'THE EIGENVALUES THAT CORRESPOND TO THE POLES  OF
      1 THE CONTINUOUS-TIME LQ CONTROLLER  ARE...'
         CALL  MEIGN(WM2,WV1,WV2,IRFM,WM1)
          END  IF
        C1=-1
        CALL MAT1(RKFSS,HM,IRFM,IRHM,IRFM,WM1)
          WRITE(KOUT,*)'  '
          WRITE(KOUT,*)'DO YOU WANT TC  CALCULATE THE POLES OF THE CONTINU
      1OUS-TIME KALMAN FILTER?   Y  OR  N>'
          READ(KIN,23)MSG
         IF(MSG.EQ.'Y') THEN
         CALL  MADD1(IRFM,IRFM,FM,WM1,WM7,C1)
         WRITE(KOUT,*)'THE EIGENVALUES  THAT ARE THE  POLES OF THE CNOTINU
      1OUS-TIME KALMAN FILTER  ARE....'
         CALL   MEIGN(WM7,WV1,WV2,IRFM,WM8)
         END IF
      CALL MADD1(IRFM,IRFM,WM2,WM1,FC,C1)
      MSG=' FC FOR THE LOG CONTCRLLER IS'
      CALL MMATIO(FC,IRFM,IRFM,IC,KIN,KCUT,NDIM,NDIM)
      WRITE(KOUT,*)' '
      WRITE(KOUT,*)'DO YOU WISH TC CALCULATE THE EIGENVALUES OF THE LOG
     1CONTROLLER   F  MATRIX?   Y  CR N>'
      READ(KIN,23)MSG
      IF (MSG.EQ.'Y') THEN
      WRITE(KOUT,*)' '
      WRITE(KOUT,*)'THE EIGENVALUES OF THE LOG CONTRCLLER F MATRIX ARE'
      CALL MEIGN(FC,WV1,WV2,IRFM,WM1)
      END IF
      RETURN
 2933 IO=0
      END
* DECK DAS1
      SUBROUTINE DAS1(GOGT,BM,V,ICBM,WM3,IRFM)
      DIMENSION GOGT(NDIM,NDIM),BM(NDIM,NDIM),V(NDIM,NDIM),
     1  WM3(NDIM,NDIM),COM1(1),COM2(1)
C    THIS SUBROUTINE MODIFIES  GOGT AND RETURNS THE MODIFIED
C     VALUE IN GOGT, WHERE GOGT IS USED IN THE KALMAN FILTER
C     GAIN CALCULATIONS.  THE MODS ARE IN ACCORCANCE WITH THE
C     THE TECHNIQUE DEVELOPED BY DCYLE AND STEIN IN "ROBUSTNESS
C      WITH OBSERVERS",IEEE TRANS. CN AUTO. CONTROL,VOIL AC24,
C     NO. 4,AUG, 79,PGS 607-611.
C
C       THE VALUE RETURNED IN GOGT IS QQ , WHERE  QQ IS
C
C    QQ=GOGT+SQ(SQ)BM(V(BMT)
C
C    SQ IS A SCALAR DESIGN PARAMETER, THAT AS IT APPROACHES
C    INFINITY,  CAUSES THE LOG CONTROLLER TO RECOVER THE ROBUSTNESS
C     PROPERTIES CF A FULL STATE FEEDBACK CONTROLLER.
CTHE MATROX--V-- IS ALSO A DESIGN, PARAMETER WITH THE REQUIREMENT
C  THAT IT BE POSITIVE DEFINITE.    BM---- IS THE CONTRCLLER MODEL INPUT
C    MATRIX.   GOGT --- IS THE CONTROLLER MODEL INPUT NCISE STRENGTH
C  MATRIX QM, PREMULTIPIED BY GM AND POST MUTIPLIED BY GMT WHERE GM IS
C    THE INPUT NOIESE MATRIX.
      CHARACTER MSG*60
      COMMCN /MAIN1/NDIM,NDIM1,COM1
      COMMCN /MAIN2/ COM2
      COMMCN /INOU/ KIN,KOUT,KPUNCH
      COMMCN /MAUM5/ MSG
      WRITE(KOUT,11)' '
      WRITE(KOUT,*)'THIS ROUTINE MODIFIES THE VALUE CF GM(QM)GMT
     1 USED IN CALCULATING THE KALMAN FILTER GAIN, RKFSS.'
      WRITE(KOUT,*)'THE MODIFIED Q IS = GM(QM)GMT+SQ*SQ(BM)V(BMT)WHERE
     1 SQ IS A SCALAR DESIGN PARAMETER AND V IS A POSITIVE DEFINITE
     1MATRIX DESIGN PARAMETER. THE LARGER SQ, THE MCRE ROBUST THE CONTRC
     1L SYSTEM WILL BE.'
       DO 5 INP=1,1000
       WRITE(KOUT,11)' '
 11    FORMAT(A10,/)
       WRITE(KOUT,*)'ENTER 1-TO INPUT SQ, 2-TO INPUT V 3- TOCALCULATE MOD
     1IFIED QQ,  4- TO EXIT THIS RCUTINE>'
       READ(KIN,*)ISEL
       GO TO (1,2,3,4)ISEL
 1     WRITE(KOUT,11)' '
       WRITE(KOUT,*)'ENTER SQ>0-->'
       READ(KIN,*)SQ
       GO TC 5
 2     WRITE(KOUT,11)' '
        WRITE (KOUT,*)'V IS INIALIZED TO .ZERO UPCN ENTRY INTO THIS OPTIO
     1N'
       DO 7 I=1,NDIM
      DO 7 J=1,NDIM
7      V(I,J)=0
      WRITE(KOUT,*)'ENTER I/O OPTION FO- POSITIVE DEF V(SEE INPUT ROUTIN
     1E)>'
      READ(KIN,*)IO
      MSG='DESIGN PARAMETER   V MATRIX ENTRIES'
      CALL MMATIO(V,ICBM,ICBM,IC,KIN,KOUT,NDIM,NDIM)
      GO TC 5
3     CALL MAT3(IRFM,ICBM,BM,V,WM3)
C  WM3=BM(V)BMT   IRFM  X  IRFM
      SQ1=SQ*SQ
      CALL MADD1(IRFM,IRFM,GOGT,WM3,V,SQ1)
C  V IRFM  X  IRFM NOW CONTAINS THE MOD. VALUE  ---QG
      MSG='THE DOYLE AND STEIN MODIFED QQ MATRIX IS'
      IO=5
      CALL MMATIO(V,IRFM,IRFM,IC,KIN,KOUT,NDIM,NDIM)
5     CONTINUE
4     CONTINUE
C   QQ IS ACCEPTABLE SO PUT INTC GCGT
      DO 20 I=1,IRFM
      DO 20 J=1,IRFM
20    GOGT(I,J)=V(I,J)
      RETURN
      END
      SUBROUTINE MYINTG(PHI,INTGA,INTBA,WME,GA,QA,FA,BA,IRFA,ICGA,IRY,
     1IRQA)
C     THIS SUBROUTINE SETS UP THE NECESSARY INTEGRALS FCR USE BY
C      THE PERFCRMANCE ANAL. ROUTINE.  THE STATE TRANSITION MATRIX,
C     EXP(FA*TIME)=EAT,  INTEG(EAT(GA) QA (GAT) EATT),  AND
C     INTEG(EAT (BA)).   WME IS A DUMMY WORK SPACE
      REAL PHI(NDIM2,NDIM2),INTGA(NDIM2,NDIM2),INTBA(NDIM2,NDIM2),
     1 WME(NDIM2,NDIM2),QA(NDIM2,NDIM2),BA(NDIM2,NDIM2),
     1GA(NDIM2,NDIM2),FA(NDIM2,NDIM2)
      DIMENSION COM1(1),COM2(1)
      COMMCN/MAIN1/NDIM,NDIM1,COM1
      COMMCN/MAIN2/ COM2
      COMMCN /INOU/ KIN,KOUT,KPUNCH
      COMMCN /RNTIM/RNTIME,DELTIM
      COMMCN /MAIN4/NDIM2,NDIM3
C***FORM GA(QA)GAT --NEED FOR KLIENMAN ROUTINE
C
      NSAV1=NDIM
      NSAV2=NDIM1
      NDIM=NDIM2
      NDIM1=NDIM2+1
      CALL DSCRT(IRFA,FA,DELTIM,INTGA,WME,10)
C WME=INT(EAT)
      CALL MAT1(WME,BA,IRFA,IRFA,IRY,INTBA)
C INTBA= INT(EAT)BA   IRFA  XIRY   NEEDED IN MXA UPDATE
      CALL MAT3(IRFA,IRQA,GA,QA,PHI)
C PHI=GA(QA)(GAT)   IRFA  X  IRFA
      CALL INTEG(IRFA,FA,PHI,INTGA,DELTIM)
C PHI=EXP(FA)   IRFA  X  IRFA
C INTGA=INTEGRAL  ( EXP(FA)(GA)(QA)(GATT)(EXP(FA)T))   IRFA  X  IRFA
C
      NDIM=NSAV1
      NDIM1=NSAV2
      RETURN
      END
* DECK DSCRTZ
      SUBRCUTINE DSCRTZ(WM1,PHIM,BCYD,BCZD,IRFM,DELTIM,FC,BCY,
    1 IRY,BCZ,IRMM,PHIT,QTD,BTD,GT,QT,FT,BT,IRFT,ICGT,ICBT,IRMT,
    1RTD,RT,GCX,ICBM)
C   THIS. ROUTINE DISCRTIZES A CONTINUOUS TIME LQG CONTROLLER USING
C  FIRST ORDER APRROXIMATIONS TO THE REQUIRED INTEGRALS
C  AND PROVIDES AN EQUIVALENT DISCRETE TIME REPRESENTATION OF THE TRUTH
```

```
C MODEL FOR USE IN THE PERFAL ROUTINE
      REAL WM1 (NDIM,NDIM),PHIM (NDIM,NDIM),BCYD (NDIM,NDIM),
     1 BCZD(NDIM,NDIM),FC(NDIM,NDIM), BCY(NDIM,NDIM),
     1 BCZ(NDIM,NDIM),PHIT(NDIM,NDIM),QTD(NDIM,NDIM),
     1 BTD(NDIM,NDIM),GT(NDIM,NDIM),QT(NDIM,NDIM),
     1 FT(NDIM,NDIM), BT(NDIM,NDIM),RTD (NDIM,NDIM),
     1 GCX(NDIM,NDIM),RT(NDIM,NDIM)
      REAL COM1(1),COM2(1)
      CHARACTER MSG*1
      COMMON/MAIN1/NDIM,NDIM1,COM1
      COMMON/MAIN2/ COM2
      COMMON/MAIN4/NDIM2,NDIM3
      COMMON /INOU/KIN,KOUT,KPUNCH
      C1=1.0
      CALL IONT(IRFM,WM1,C1)
      CALL MA001(IRFM,IRFM,WM1,FC,PHIM,DELTIM)
C PHIM=I+FC(DELTIM) IST ORDER APPROX TO STATE TRANS MATRIX OF CONT
C CALCULATE GCX =-GCSTR
        C1=-1.0
C  RECALL THAT GCSTR WAS PASSED INTO THIS ROUTINE IN BTD
      CALL MSCALE(GCX,BTD,ICBM,IRFM,C1)
       C1=1.0
      CALL MSCALE(BCYD,BCY,IRFM,IRY,DELTIM)
C  BCYD DISCRETE TIME APPROX OF BCY
      CALL MSCALE(BCZD,BCZ,IRFM,IRFM,DELTIM)
C  BCZD DISCRETE TIME APPROX OF BCZ
      NSAV3=NDIM2
      NDIM2=NDIM
      CALL MYINTG(PHIT,QTD,BTD,WM1,GT,QT,FT,BT,IRFT,ICGT,ICBT,ICGT)
      NDIM2=NSAV3
C PHIT,QTD,BTD ARE EQUIV. DISCRETE TIME REPRESTATIONS OF TRUTH MODL
C MATRICE:
      WRITE(KOUT,*)' '
      WRITE(KOUT,*)' WAS THE VALUE ENTERED IN RT DURING INPUT A CONTINUO
     1US TIME  OR A DISCRETE TIME VALUE?  ENTER A C FOR  CONTINUOUS , A
     1 D  FOR DISCRETE VALUE>'
      READ(KIN,12)MSG
 12   FORMAT(A1)
      IF (MSG.EQ.'C') THEN
      C1=1/DELTIM
      CALL MSCALE(RTD,RT,IRMT,IRMT,C1)
      ELSE
      CALL EQUATE(RTD,RT,IRMT,IRMT)
      END IF
C RTD IS THE DISCRETE TIME APPROX OF RT
      C1=1.0
      CALL IONT(IRFT,WM1,C1)
C  WM1=GTD  = I
      END
* DECK DLQGRS
      SUBROUTINE DLQGRS(GCX,GCY,GCZ,BCY,BCZ,PHIT,PHIC,RTD,GTD,QTD,
     1 BTD,FM,BM,QM,GM,RM,HM,GT,QT,FT,BT,RT,HT,WXX,WUU,GCSTR,RKFSS,YD,
     1 IRY,IFLGCZ,WXU,WM1,WM2,WM3,WM4,WM5,WV1,WV2)
CTHIS  SUBROUTINE FORMATS THE SAMPLED DATA CONTROLLER INTO THE FORMAT
C  REQUIRED BY THE PERFORMANCE ANALYSIS ROUTINE
C THE FORMAT IS SPECIFIED IN THE COMMENT STATEMENTS IN THE CODE, AND
C IN MORE DETAIL IN E. LLOYD S MASTERS THESIS,D81,AFIT.
      REAL COM1(1),COM2(1)
      REAL GCX(NDIM,NDIM),GCY(NDIM,NDIM),GCZ(NDIM,NDIM),
     1BCY(NDIM,NDIM), BCZ(NDIM,NDIM),PHIT(NDIM,NDIM),
     1 PHIC(NDIM,NDIM),RTD(NDIM,NDIM),GTD(NDIM,NDIM),
     1QTD(NDIM,NDIM),BTD(NDIM,NDIM),FM(NDIM,NDIM),
     1 BM(NDIM,NDIM), QM(NDIM,NDIM),GM(NDIM,NDIM),
     1 WV1(NDIM),WV2(NDIM),
     1 RM(NDIM,NDIM),HM(NDIM,NDIM),GT(NDIM,NDIM),
     1 QT(NDIM,NDIM), FT(NDIM,NDIM), BT(NDIM,NDIM),
     1 HT(NDIM,NDIM),FT(NDIM,NDIM),WXX(NDIM,NDIM),
     1 WUU(NDIM,NDIM),GCSTR(NDIM,NDIM), RKFSS(NDIM,NDIM),
     1 WXU(NDIM,NDIM),WM1(NDIM,NDIM),WM2(NDIM,NDIM),
     1 WM3(NDIM,NDIM),WM4(NDIM,NDIM),WM5(NDIM,NDIM)
      REAL YO(NDIM3)
      INTEGER IFLGCZ
      CHARACTERMSG*60,MSG1*1,MSG2*1
      COMMCN /MAIN1/NDIM,NDIM1,COM1
      COMMCN /MAIN2/COM2
      COMMCN/INCU/KIN,KCUT,KPUNCH
      COMMCN /MAIN4/NDIM2,NDIM3
      COMMCN/MAUN5/MSG
      COMMCN /RNTIM/RNTIME,DELTIM
      COMMCN/MAIN6/ICBT,ICBM,ICFA,ICGA,ICGM,ICGT,ICQA,IRFA,IRFM,IRFT,
     1 IRHT,IRQA,IO,LOG,IRHM,NUMOTS
      WRITE(KOUT,*)'COLORED INPUT NOISE (Y OR N) >'
      READ(KIN,11)MSG
      IHOLD=IRFM                                                          — A
      IF(MSG.EQ.'N')GO TO 25
      CALL CNOISE(FM,GM,HM,BM,QM,IRFM,IRHM,ICGM,ICBM,WM1,WM2,WM3)
 25   WRITE(KOUT,*)'ENTER THE TOTAL RUN TIME AND SAMPLE TIME>'
      READ(KIN,*)RNTIME,DELTIM
 11     FORMAT(A1)
C CALCULATE EQUIVALENT DISCRETE TIME VERSICNS OF, BM--BMO,GM--GMO=I,
C  GM--QMO,   AND PHIM THE STATE TRANSITICN MATRIX FCR FM
C**********************************************************************
C  SINCE WORKSPACE IS AT A PREMIUM,  THE TRUTH MAODEL MATRICES
CPHIT,BTD,GTD,QTD,RTD WILL BE USED FOR THEIR  CONTROLLER MODEL
C COUNTER PARTS DURING THIS ROUTINE BEFORE THE EQUIVALENT DISCRETE
C TIME TRUTH MOCEL IS COMPUTED.  AT THAT TIME THERE IS NO LONGEG ANY
C NEED FOR THOSE CONTROLLER MODEL MATRICE SINCE THE CONTROLLER IS PUT
C INTO THE PERFCMANNCE ANALYSIS FCRMAT,PHIC BCY,.....GCZ
C*******************************************************************
      NSAV3=NDIM2
      NDIM2=NDIM
      CALL MYINTG(PHIT,QTD,BTD,RTC,GM,QM,FM,BM,IRFM,ICGM,ICBM,ICGM)
C RTD IS USED AS DUMMY WORK SPACE IN CALL TO MYINTG
      NDIM2=NSAV3
      CALL IDNT(IRFM,GTD,1.000)
      WRITE(KOUT,*)'ENTER A C IF THE  VALUED ENTERED INTO RM IS A CONTIN
     1UOUS TIME VALUE TO FORM THE BASIS OF AN APPROXIMATE DISCRETE TIME
     1 RM,ENTER A  D  OTHERWISE>'
      READ(KIN,11)MSG1
      IF   (MSG1.EQ.'C') THEN
C   APPROXIMATE RMO=RM/SAMPLE TIME
      C1=1/DELTIM
      CALL MSCALE(RTD,RM,IRHM,IRHM,C1)
      ELSE
C  THE VALUE IN RM IS DISCRETE TIME ALREADY
      CALL EQUATE(RTD,RM,IRHM,IRHM)
      END IF
C SET UP X , S , AND U FOR DOTCON
      CALL XSU(GCZ,GCY,PHIC,GCX,BCY,BCZ,RKFSS,WM1,WM2,WM3,WM4,WM5,        — A
     1 FM,BM,IHOLD,ICBM,WXX,WUU,WXU,PHIT)
C GCZ NOW CONTAINS X, PHIC CONTAINS U,GCY CONTAINS  S
C GCX,BCZ,BCY,RKFSS WERE DUMMY WORK AREAS IN XSU
      CALL DOTCON(PHIT,WXX,GCX,GCZ,BTD,PHIC,RKFSS,BCY,BCZ,GCY,WUU,
     1GCSTR,IHOLD)
      IF(IHOLD.EQ.IRFM) GO TO 35
      ICH=IHOLD+1                                                         — A,C
      DO 30 I=I,ICBM
      DO 30 J=ICH,IRFM
 30   GCSTR(I,J)=0.
 35   CONTINUE
CBCY,BCZ,GCY,PHIC,GCZ,GCX,RKFSS  ARE USED AS DUMMY WORK SPACE IN DOTCON
      WRITE(KOUT,*)'DO YOU WISH TO COMPUTE THE KALMAN FILTER GAIN OF PIC
     1K IT DIRECTLY (AS IN MAYBECK SECTION, 14.5)ENTER A C TO COMPUTE
     1NTER A P TO PICK IT DIRECTLY>'
      READ (KIN,11)MSG2
      IF (MSG2.EQ.'C')THEN
      CALL DKFTR(PHIT,BTD,GTD,QTD,GCX,RTD,HM,GCY,GCZ,PHIC,RKFSS,BCY,BCZ,
     1FM,GM,QM,BM,WM1,WM2)
CGCX,GCY,GCZ,BCZ,BCY ARE USED AS DUMMY WORK SPACE IN CALL TO DKFTR
      ELSE
      CALL PKDIRC(GCX,PHIT,GCY,GCZ,HM,BTD,RKFSS,ICBM,IRFM,IRHM)
C   GCX,GCY,GCZ ARE, DUMMY WORKSPACES IN CALL TO PKDIRC
      END IF
C************INTHE FOLLOWING CALCULATIONS,  BCY,GCY,BCZ ARE DUMMY WORKSP
C***********ACES UNTIL THEIR LAST USE WHEN THEY ARE SET EQUAL TO THEIR F
C*********INAL VALUES FOR PERFAL SUBROUTINE
      C1=1.0
      CALL IDNT(IRFM,BCY,C1)
      CALL MAT1(RKFSS,HM,IRFM,IRHM,IRFM,GCY)
      C1=-1.0
      CALL MADD1(IRFM,IRFM,BCY,GCY,BCZ,C1)
C BCZ = I-RKFSS(HM)
       WRITE(KOUT,*)' '
       WRITE(KOUT,*)'DO YOU WISH TO CALCULATE  THE SAMPLED-DATA FILTER P
     1OLES?   Y   OR   N>'
       READ(KIN,12)MSG
       IF (MSG.EQ.'Y') THEN
       CALL MAT1(PHIT,BCZ,IRFM,IRFM,IRFM,PHIC)
       WRITE(KOUT,*)'THE EIGENVALUES THAT CORRESPOND TO THE SAMPLED-DATA
     1 FILTER POLES ARE.....'
        CALL MEIGN(PHIC,WV1,WV2,IRFM,WM1)
          END IF
      CALL MSCALE(BCY,GCSTR,ICBM,IRFM,C1)
C BCY=-GCSTR     ICBM   X   IRFM
      IF (MSG2.EQ.'C') THEN
C  FORMULATE THE OPTIMAL CONTROL LAW FOR PERFAL
      CALL MAT1(BCY,BCZ,ICBM,IRFM,IRFM,GCX)
C   GCX= -GCSTR(I-RKFSS(HM))   ICBM   X   IFRM
      CALL MAT1(BCY,RKFSS,ICBM,IRFM,IRHM,GCZ)
C   GCZ=  -GCSTR(RKFSS)  ICBM  X  IRHM
      IFLGCZ=0
      CALL  MAT1(BTD,BCY,IRFM,ICBM,IRFM,GCSTR)
      C1=1.0
      CALL MADD1(IRFM,IRFM,PHIT,GCSTR,BCY,C1)
C   BCY=  PHIT-BTD(GCTR)    IRFM  X  IRFM
       WRITE(KOUT,*)' '
       WRITE(KOUT,*)'DO YOU WISH TO CALCULATE  THE POLES OF  THE OPTIMAL
     1 LQ SAMPLED-DATA CONTROLLER?      Y  OR  N>'
       READ(KIN,12)MSG
       IF (MSG.EQ.'Y')  THEN
       WRITE(KOUT,*)'THE EIGENVALUES THAT CORRESPOND TO THE  POLES OF TH
     1E OPTIMAL  LQ CONTROLLER ARE.........'
       CALL MEIGN(BCY,WV1,WV2,IRFM,PHIC)
       END  IF
      CALL MAT1(BCY,BCZ,IRFM,IRFM,IRFM,PHIC)
C   PHIC =(PHIT-BTD(GCSTR))(I-RKFSS(HM))  IRFM  X  IRFM
      CALL MAT1(BCY,RKFSS,IRFM,IRFM,IRHM,BCZ)
C   BCZ=(PHIT-BTD(GCSTR))RKFSS  IRFM  X  IRHM
      ELSE
C   FORMULATE THE SUBOPTIMAL CONTROL LAW USTR=-GCSTR(X AT  TSUB I MINUS)
      CALL EQUATE(GCX,BCY,ICBM,IRFM)
C   GCX=-GCSTR  ICBM  X  IRFM
      CALL MAT1(PHIT,BCZ,IRFM,IRFM,IRFM,GCSTR)
      C1=1.0
      CALL MAT1(BTD,GCX,IRFM,ICBM,IRFM,BCY)
       WRITE(KOUT,*)' '
       WRITE(KOUT,*)'DO YOU  WISH TO  CALCULATE THE  POLES OF THE OPTIMA
     1L  LQ  SAMPLED DATA CONTROLLER?   Y   OR  N>'
```

```
      READ(KIN,12)MSG
      IF (MSG.EQ.'Y') THEN
       CALL MADD1(IRFM,IRFM,PHIT,BCY,PHIC,C1)
      WRITE(KOUT,*)'THE EIGENVALUES THAT CORRESPOND TO THE POLES OF THE
     1OPTIMAL  LQ CONTROLLER ARE........'
       CALL MEIGN(PHIC,WV1,WV2,IRFM,WM1)
      END IF
      CALL MADD1(IRFM,IRFM,GCSTR,BCY,PHIC,C1)
C  PHIC= PHIT(I-RKFSS(HM))-BTD,GCSTR  IRFM  X  IRFM
      CALL MAT1(PHIT,RKFSS,IRFM,IRFM,IRHM,BCZ)
C  BCZ=PHIT(RKFSS)  IRFM   X  IRHM
      DO 101 I=1,IRFM
      DO 101J=1,IRHM
101   GCZ(I,J)=0
      IFLGCZ=1
      END IF
      IRY=1
      DO 102 I=1,NDIM
      DO 102 J=1,IRY
      GCY(I,J)=0
102      BCY(I,J)=0
      DO 107 I=1,1000
107      YD(I)=0
      IO=5
      MSG='PHIC FOR THE SAMPLED DATA CONTROLLER IS'
      CALL MMATIO(PHIC,IRFM,IRFM,IO,KIN,KOUT,NDIM,NDIM)
      WRITE(KOUT,*)' '
      WRITE(KOUT,*)'DO YOU WANT TO CALCULATE THE EIGENVALUES OF THE LQG
     1 CONTROLLER STATE TRANSITION MATRIX, PHIC?  Y  OR  N>'
      READ(KIN,12)MSG
      IF (MSG.EQ.'Y') THEN
      WRITE(KOUT,*)'THE EIGENVALUES OF THE LQG CONTROLLER STATE TRANSITI
     1ON MATRIX ARE.....'
      CALL MEIGN(PHIC,WV1,WV2,IRFM,WM1)
      END IF
      MSG='BCZ FOLLOWS, BCY=0'
      CALL MMATIO(BCZ,IRFM,IRHM,IC,KIN,KOUT,NDIM,NDIM)
      MSG='GCX FOR THE SAMPLED DATA CONTROLLER IS'
      CALL MMATIO(GCX,ICBM,IRFM,IC,KIN,KOUT,NDIM,NDIM)
      IF (MSG2.EQ.'C') THEN
      MSG='GCZ FOR A COMPUTED KFSS'
      ELSE
      MSG='GCZ FOR KFSS PICKED DIRECTLY'
      END IF
      CALL MMATIO(GCZ,ICBM,IRHM,IC,KIN,KOUT,NDIM,NDIM)
      WRITE(KOUT,*)' GCY IS SET = 0'
C  THIS IS A REGULATOR SO YD IS ALWAYS ZERO
      NSAV3=NDIM2
      NDIM2=NDIM
      CALL MYINTG(PHIT,QTD,BTD,RTD,GT,QT,FT,BT,IRFT,ICGT,ICBT,ICGT)
C  RTD IS USED AS DUMMY WORK SPACE IN CAL TO MYINTG
      NDIM2=NSAV3
C PHIT,QTD,BTD ARE EQUIV. DISCRETE TIME REPRESTATIONS OF TRUTH MODL
C MATRICE
      WRITE(KOUT,*)' '
      WRITE(KOUT,*)' WAS THE VALUE ENTERED IN RT DURING INPUT A CONTINUO
     1US TIME  OR A DISCRETE TIME VALUE?  ENTER A C FOR  CONTINUOUS , A
     1 D  FOR DISCRETE VALUE>'
      READ(KIN,12)MSG
12    FORMAT(A1)
      IF (MSG.EQ.'C') THEN
      C1=1/DELTIM
      CALL MSCALE(RTD,RT,IRHT,IRHT,C1)
      ELSE
      CALL EQUATE(RTD,RT,IRHT,IRHT)
      END IF
```

```
C RTD IS THE DISCRETE TIME APPROX OF RT
      C1=1.0
      CALL IDNT(IRFT,GTD,C1)
C  GTD=GTD  = I
      RETURN
      END
* DECK DKFTR
      SUBROUTINE DKFTR(PHIM,BMD,GMD,QMD,WM1,RMD,HM,WM3,WM5,WM6,RKFSS,
    1 F2,H2,FM,GM,QM,BM,WM2,WM4)

C       THIS ROUTINE CALCULATE THE STEADY STATE KALMAN FILTER GAIN MATRIX
C       FOR A SAMPLED  ATA CONTROLLER
      REAL PHIM(NDIM,NDIM),BMD(NDIM,NDIM),QMD(NDIM,NDIM),
    1 GMD(NDIM,NDIM),WM1(NDIM,NDIM),RMD(NDIM,NDIM),
    1 HM(NDIM,NDIM), WM3(NDIM,NDIM),   WM5(NDIM,NDIM),
    1 WM6(NDIM,NDIM),  RKFSS(NDIM,NDIM),F2(NDIM,NDIM)
    1 ,H2(NDIM,NDIM),FM(NDIM,NDIM),
    1 GM(NDIM,NDIM),QM(NDIM,NDIM),BM(NDIM,NDIM),
    1 WM2(NDIM,NDIM),WM4(NDIM,NDIM)


      REAL COM1(1),COM2(1)
      COMMON /MAIN1/NDIM,NDIM1,COM1
      COMMON /INOU/ KIN,KOUT,KPUNCH
      COMMON /RATIM/ RNTIME,DELTIM
      COMMON/MAIN4/NDIM2,NDIM3
      COMMON/MAUN5/MSG
      COMMON /MAIN2/ COM2
       COMMON /MAIN6/ICET,ICBM,ICFA,ICGA,ICGM,ICGT,ICQA,IRFA,IRFM,IRFT,
    1 IRHT,IRQA,IO,LQG,IRHM,NUMDTS
      CHARACTER MSG1*1,MSG*60
C  DELETE DETERMINISTIC STATES AS IN THE CONTINUOUS TIME CASE
      WRITE(KOUT,*)'IF YOU PLAN TO USE THE DOYLE AND STEIN TECHNIQUE FOR
    1 THIS RUN YOU MAY WISH TO MODIFY THE VALUE OF NUMDTS, THE NUMBER O
    1F DETERMINISTIC STATES.  DO YOU WANT TO CHANGE NUMDTS?  Y  OR  N>'
      READ(KIN,11)MSG1
      NUMSAV=NUMDTS
      IF (MSG1.EQ.'Y') THEN
      WRITE(KOUT,*)'ENTER THE NEW VALUE OF NUMDTS FOR THIS RUN>'
      READ(KIN,*)NUMDTS
      END IF
      WRITE(KOUT,*)'NUMDTS= ',NUMDTS
      IDS=NUMDTS+1
      IRF2=IRFM-NUMDTS
      IF(NUMDTS.EQ.0) THEN
C  STORE SYSTEM MODEL IN INTERMEDIATE MATRICES COMPATIBBLE
C  WITH THOSE BELOW WHEN THER ARE DETERMINISTIC STATES REMOVED
      CALL EQUATE(F2,PHIM,IRF2,IRF2)
      CALL EQUATE(WM1,GMD,IRF2,IRF2)
      CALL EQUATE(H2,HM,IRHM,IRF2)
      CALL EQUATE(WM5,BMD,IRF2,ICBM)
      CALL EQUATE(WM2,QMD,IRF2,IRF2)
      ELSE
C DELETE THE DETERMINISTIC STATES FORM THE MODEL USED TO FORM
C  THE STEADY STATE KALMAN FILTER GAIN MATRIX
      DO 2112 I=IDS,IRFM
      II=I-NUMDTS
      DO 2112 J=IDS,IRFM
      JJ=J-NUMDTS
2112  WM4(II,JJ)=FM(I,J)
      DO 2113 I=1,IRHM
      DO 2113 J=IDS,IRFM
      JJ=J-NUMDTS
2113  H2(I,JJ)=HM(I,J)
C FORM B2D G2D NOW
      DO 2115 I=IDS,IRFM
```

```
          II=I-NUMDTS
          DO 2115 J=1,ICBM
2115      WM3(II,J)=BM(I,J)
          DO 2114 I=IDS,IRFM
          II=I-NUMDTS
          DO 2114 J=1,ICGM
2114      WM1(II,J)=GM(I,J)
11           FORMAT(A1)
          NSAV=NDIM2
          NDIM2=NDIM
          CALL MYINTG(F2,WM2,WM5,WM6,WM1,QM,WM4,WM3,IRF2,ICGM,ICBM,ICGM)
          NDIM2=NSAV
          C1=1.0
          CALL IDNT(IRF2,WM1,C1)
          END IF
C CALCULATE GMD (QMD) GMDT
          CALL MAT3(IRF2,IRF2,WM1,WM2,RKFSS)
CRKFSS IS DUMMY WORK SPACE AT THIS POINT IN THE PROGRAM
          WRITE(KOUT,*)' DO YOU WISH TO MODIFY THE QMD MATRIX BY THE DOYLE
        1 AND STEIN TECHNIQUE FOR CONTINUOUS TIME CONTROLLERS EXTENDED TO
        1 THE DISCRETE TIME SYSTEMS,  Y  OR  N>'
          READ(KIN,11)MSG1
          IF (MSG1.EQ.'Y') THEN
          CALL DAS2(BM(IDS,IDS),RKFSS,WM1,ICBM,WM3,IRF2,WM6)
C  RETURNS A MODIFIED QMD VALUE TO BE USED IN FINDING RKFSS
          END IF
C   CALCULATE THE KALMAN FILTER GAINS, RKFSS, FOR EITHER THE MODIFIED
C   QMD OR THE UNMODIFIED QMD
C    QMD IS STORED IN RKFSS
          CALL TRANS2(IRHM,IRF2,H2,WM3)
          DO 101 I=1,IRHM
101       WM4(I,1)=RMD(I,I)
          CALL KFLTR(IRF2,IRHM,F2,WM3,RKFSS,WM4,WM6,WM1,WM5)
C  WM6=PMSS,WM5 CLOSED LOOP MEAS MATRIX
C WM1 = RKFSS WITHOUT THE ZEROS FOR THE DETERMINISTIC STATES
C   NOW ADD THE ZEROS FOR THOSE STATES
          IF (NUMDTS.EQ.0)THEN
          DO 2029 I=1,IRFM
          DO 2029J=1,IRHM
2029      RKFSS(I,J)=WM1(I,J)
          ELSE
          DO 2119 J=1,IRHM
          DO 2118I=1,NUMDTS
2118      RKFSS(I,J)=0
          DO 2119 I=IDS,IRFM
          II=I-NUMDTS
2119      RKFSS(I,J)=WM1(II,J)
          END IF
C  NOW WRITE OUT THE RKFSS MATRIX
27           IO=5
          MSG='STEADY STATE SAMPLED DATA KALMAN FILTER GAIN MATRIX'
          CALL MMATIO(RKFSS,IRFM,IRHM,IO,KIN,KOUT,NDIM,NDIM)
          NUMDTS=NUMSAV
          RETURN
           END
* DECK DAS2
          SUBROUTINE DAS2(BM,QMD,V,ICBM,WM3,IRFM,WM1)
          REAL BM(NDIM,NDIM),QMD(NDIM,NDIM), V(NDIM,NDIM),
        1 WM3(NDIM,NDIM),  WM1(NDIM,NDIM)
C  THIS ROUTINE ALLOWS QMD TO  BE MODIFIED IN A MANNER SIMILAR TO THE DO
C   YLE AND STEIN TECHNIQUE FOR CONTINUOUS TIME SYSTEMS.
C  QMDMOD=QMD +(SQ*SQ)(BM(V)BMT)*SAMPLE  TIME
C  WHERE SQ IS A SCALAR DESIGN PARAMETER AND V  IS A POSITIVE DEFINITE
C SYMMETRIC MATRIX  DESIGN PARAMETER.    AS  SQ--->TO INFINITY IN THE CO
C NTINUOUS TIME CASE,  ROBUSTNESS PROPERTIES OF A FULL STATE  FEEDBACK
C  CONTROLLER ARE RECOVERED.  THIS ROUTINE IS BASED ON E. LLOYDS MASTERS
```

```
C  THESIS ,D81,  AFIT
      CHARACTER MSG*60
      REAL COM1(1),COM2(1)
      COMMON /MAIN1/NDIM,NDIM1,COM1
      COMMON /RNTIM/RNTIME,DELTIM
      COMMON /MAUN5/MSG
      COMMON /MAIN2/ COM2
      COMMON /INOU/KIN,KOUT,KPUNCH
      WRITE(KOUT,11)' '
 11   FORMAT(A1)
      WRITE(KOUT,*)'THIS ROUTINE MODIFIES THE VALUE OF QMD USED TO CALCU
     1LATE THE STEADY STATE KALMAN FILTER GAIN.  THE MODIFICATION PERFOR
     1MED IS SIMILAR TO THE DOYLE AND STEIN TECHNIQUE FOR CONTINUOUS-TIM
     1E SYSTEMS.  FOR A COMPLETE DESCRIPTION SEE E. LLOYDS MASTERS THESI
     1S, AFIT, D81.  BRIEFLY THE MODIFICATION ATTEMPTS TO ENHANCE ROBUST
     1NESS OF THE LQG CONTROLLER AND IS   QMOD=GMD(GMD)GMDT+SQ*SQ*DELTIM
     1 *(BM(V)BMT))...  THE LARGER THE VALUE CHOSEN FOR THE SCALAR SQ TH
     1E MORE ROBUSTNESS RECOVERY (COMPARED TO FULLSTATE FEED-BACK),  V M
     1UST BE A POSITIVE DEF. MATRIX.  CHOOSING V=I ADDS PSEUDONOISE EQUA
     1LLY TO ALL CONTROL INPUTS.'
      WRITE(KOUT,11)' '
      WRITE(KOUT,*)'ENTER 1-- TO INPUT SQ, 2-- TO INPUT V,  3-- TO
     1 COMPUTE MODIFIED Q, 4-- TO EXIT ROUTINE.......NOTE 1 ,2 MUST BE
     1 ACCOMPLISHED BEFORE 3,  AND 3 BEFORE 4,  BUT THAT 1,2,3, CAN BE
     1 DONE ANY NUMBER OF TIMES BEFORE USING 4'
 5    WRITE(KOUT,11)' '
      WRITE(KOUT,*)'ENTER OPTION>'
      READ(KIN,*)IOPT
      GO TO ( 1,2,3,4)IOPT
 1    WRITE(KOUT,11)' '
      WRITE(KOUT,*)'ENTER SQ>'
      READ(KIN,*)SQ
      GO TO 5
 2    WRITE(KOUT,11)' '
      WRITE(KOUT,*)'V IS INITIALIZED TO THE IDENTITY MATRIX UPON ENTRY T
     1O THIS OPTION.  IF YOU DESIRE TO CHANGE V ...REMEMBER IT MUST BE
     1 POSITIVE DEFINITE..... ENTER THE I/O OPTION (I/O OPTIONS ARE
     1 PRINTED AT THE BEGINNING OF THE PROGRAM) ELSE ENTER A 0  >'
      C1=1.0
      CALL IDNT(ICBM,V,C1)
      READ(KIN,*)IO
      IF (IO.EQ.0) THEN
      GO TO 5
      ELSE
      MSG='THE CHOSEN V MATRIX IS'
      CALL MMATIO(V,ICBM,ICBM,IO,KIN,KOUT,NDIM,NDIM)
      END IF
      GO TO 5
 3    WRITE(KOUT,11)' '
      CALL MAT4(V,BM,ICBM,ICBM,IRFM,WM3)
      CALL MAT1(BM,WM3,IRFM,ICBM,IRFM,WM1)
      C1=SQ*SQ*DELTIM
      CALL MADD1(IRFM,IRFM,QMD,WM1,WM3,C1)
      IO=5
      MSG='MODIFIED Q MATRIX, QMOD='
      CALL MMATIO(WM3,IRFM,IRFM,IO,KIN,KOUT,NDIM,NDIM)
      GO TO 5
 4    CALL EQUATE(QMD,WM3,IRFM,IRFM)
C  REPLACE THE VALUE IN QMD WITH QMOD
      RETURN
      END
*DECK DOTCON
      SUBROUTINE DOTCON(PHIM,WXX,WM1,X,BMD,U,WM2,PHIPRM,
     1 XPRIM,S,WUU,GCSTR,IHOLD)                                   A
      REAL COM1(1),COM2(1), PHIM(NDIM,NDIM),
     1WXX (NDIM,NDIM),WM1(NDIM,NDIM),X(NDIM,NDIM),
     1 BMD(NDIM,NDIM), U(NDIM,NDIM),WM2(NDIM,NDIM),
     1 PHIPRM(NDIM,NDIM),XPRIM(NDIM,NDIM),S(NDIM,NDIM),
     1 WUU(NDIM,NDIM),GCSTR(NDIM,NDIM)
      CHARACTER MSG*60
      COMMON /MAIN1/NDIM,NDIM1,COM1
      COMMON /MAIN2/ COM2
      COMMON /MAIN4/NDIM2,NDIM3
      COMMON /INOU/KIN,KOUT,KPUNCH
      COMMON /RNTIM/ RNTIME,DELTIM
      COMMON /MAIN6/ICST,ICBM,ICFA,ICGA,ICGM,ICGT,ICOA,IRFA,IRFM,IPFT,
     1 IRMT,IRQA,IO,LOG,IRMM,NUMOTS
      COMMON /MAUN5/MSG
C THIS SUBROUTINE COMPUTES THE STESDY STATE  OPTIMAL FEEDBACK GAIN MATRI
CX  GCSTR,  BASED ON A LINEAR QUADRATIC COST CRITERION, FOR A SAMPLED
C  DATA CONTROLLER
C  SEE MAYBECK CHAP 14 FOR A DETAILED DISCUSSION OF ALGORITHM AND
C EQUATIONS
C*****************************************************************
      MT=1
C
C  TRANSFORM SYSTEM SO KLEINMAN RICCATI SOLVER WILL HANDLE S NOT=0
      CALL PRIMIT(WM2,U,ICBM,GCSTR,S,IHOLD,BMD,WM1,PHIPRM,X,XPRIM,      — A
     1 PHIM)
C
C WM2=UI*ST
CNOW COMPUTE KPRIM FROM RICCATI EQUATION
      CALL MAT3(IHOLD,ICBM,BMD,GCSTR,WM1)
      CALL DRIC(IHOLD,PHIPRM,WM1,XPRIM,X,GCSTR)                         — A
C GCSTR CONTAINS INFO THAT IS NOT USED
C X CONTAINS **KPRIM**  IRFM  X  IRFM
C  NOW COMPUTE GCSTRPRIM
      C1=1.0
      CALL MAT3A(ICBM,IHOLD,BMD,X,GCSTR)                                — A,C
      CALL MADD1(ICBM,ICBM,U,GCSTR,WM1,C1)
      CALL  GPINV(ICBM,ICBM,WM1,GCSTR,MR,MT)
      IF (MR.NE.ICBM) THEN
      PRINT *,'INVERSE OF U IN DDTCCN NOT OF FULL RANK,  RANK IS ',
     1 MR,' RANK SHOULD BE =ICBM=',ICBM
      END IF
      CALL MAT4(GCSTR,BMD,ICBM,ICBM,IHOLD,WM1)
      CALL MAT1(WM1,X,ICBM,IHOLD,IHOLD,GCSTR)                           — A
      CALL MAT1(GCSTR,PHIPRM,ICBM,IHOLD,IHOLD,WM1)
C WM1=GCSTRPRIM=((U+BMDT(KPRIM)BMD)I)(BMD(KPRIM)PHIPRM)
C   ICBM   X   IRFM
      CALL MADD1(ICBM,IHOLD,WM1,WM2,GCSTR,C1)                           — A
C  GCSTR ICBM   X   ICBM
      IO =5
      MSG='THE OPTIMAL STEADY STATE FEEDBACK GAIN MATRIX,GCSTR'
      CALL MMATIO(GCSTR,ICBM,IHOLD,IO,KIN,KOUT,NDIM,NDIM)               — A
      RETURN
      END
* DECK PKDIRC
      SUBROUTINE PKDIRC(W,PHIM,WM1,WM2,HM,BMD,RKFSS,ICBM,IRFM,IRHM)
      REAL COM1(1),COM2(1),W(NDIM,NDIM),
     1 PHIM (NDIM,NDIM), WM1(NDIM,NDIM), WM2(NDIM,NDIM),
     1 HM(NDIM,NDIM), BMD(NDIM,NDIM), RKFSS(NDIM,NDIM)
      CHARACTER MSG*60,MSG1*1
      COMMON /MAIN1/NDIM,NDIM1,COM1
      COMMON /MAIN2/COM2
      COMMON /MAUN5/ MSG
      COMMON /INOU/ KIN, KOUT,KPUNCH
C  AS IN MAYBECK, SECTION 14.5, RKFSS=PHIMI(BMD)W *SQ.  SQ IS A
C SCALAR DESIGN PARAMETER AND W IS ANY NONSINGULAR  M X M MATRIX.
C  MAYBECK SUGGESTS THAT W=(HM(PHIMI)BMD)I IS A POSSIBLE CHOICE.
C  THE RKFSS PICKED AS A RESULT OF THIS  ALGORITHM FORMS THE BASIS
C OF A SUBOPTIMAL CONTROL LAW,USTAR=GCSTR(X(TI-MINUS)
      WRITE(KOUT,12) ' '
      WRITE(KOUT,*)'THIS ROUTINE CALCULATES THE STEADY STATE KALMAN FILT
     1ER GAIN DIRECTLY( THAT IS WITH OUT USE PMSS FROM THE MATRIX RICCAT
     1I EQUATION AS THE BASIS OF KFSS)I EQUATION AS IN SECTION 14.5 .MAY
     1BECK.  KFSS=SQ*(PHIMI)BMD(W)  WHERE THE SCALAR SQ AND THE MATRIX
     1 W   ARE DESIGN PARAMETERS...THE LARGER THE SQ THE MORE ROBUSTNESS
     1  THE SUBSEQUENT CONTROLLER WILL HAVE.  NOTE THERE ARE NO STABILIT
     1Y CLAIMS FOR THE RESULTING CONTROLLER.  SO BE SURE TO CHECK THE EI
     1GENVALUES OF THE SUBSEQUENT CONTROLLER.'
      WRITE(KOUT,12)' '
12     FORMAT(A1/)
      WRITE(KOUT,*)'THE OPTIONS FOR THIS ROUTINE ARE 1->CHOOSE SQ, 2-> C
     1HOOSE W, 3-. COMPUTE AND PRINT RKFSS, 4-> EXIT ROUTINE........'
5      WRITE(KOUT,12)' '
       WRITE(KOUT,*)'ENTER OPTION>'
      READ(KIN,*)IOPT
      GO TO (1,2,3,4)IOPT
      GO TO 4
1     WRITE(KOUT,12)' '
      WRITE(KOUT,*)'ENTER A VALUE FOR SQ. LARGER SQ GIVE BETTER ROBUSTNE
     1SS>'
      READ(KIN,*)SQ
      GO TO 5
2      WRITE(KOUT,12)' '
      WRITE(KOUT,*)'DO YOU WANT TO PICK W ARBTRARILY OR DO YOU WANT W TO
     1 BE HM(PHIMI)BMD --INVERSE AS IN MAYBECK SECTION 14.5. NOTE THAT
     1 IRHM MUST BE EQUAL TO ICBM SINCE W MUST BE ICBM  X  ICBM.  ENTER
     1AN  A  FOR ARBITRARY,  W  OTHERWISE>'
      READ(KIN,11)MSG1
11    FORMAT(A1)
      IF(MSG1.EQ.'A')THEN
      WRITE(KOUT,*)'ENTER I/O OPTION FOR W(SEE INPUT ROUTINE FOR EXPLANA
     1TION OF INPUT  OPTIONS 1,2,3,4,5,6)>'
       READ(KIN,*)IO
      MSG='ARBITRARY W MATRIX'
      CALL MMATIO(W,ICBM,ICBM,IO,KIN,KOUT,NDIM,NDIM)
      ELSE
C COMPUTE W AS DESCRIBED ABOVE
      IF (ICBM.NE.IRHM) THEN
      WRITE(KOUT,*)'NOTE THAT IRHM MUST EQUAL ICBM TO USE THIS METHOD OF
     1 CALCULATING W, ICBM=',ICBM,'  IRHM= ' ,IRHM
      GO TO 5
       END IF
      CALL EQUATE(WM1,PHIM,IRFM,IRFM)
C  GMINV DESTROYS THE CALLING ARRAY
      MT=1
      CALL GMINV(IRFM,IRFM,WM1,WM2,MR,MT)
      CALL MAT1(HM,WM2,IRHM,IRFM,IRFM,WM1)
      CALL MAT1(WM1,BMD,IRHM,IRFM,ICBM,WM2)
      CALL GMINV(IRHM,ICBM,WM2,W,MR,MT)
      END IF
      GO TO 5
3      WRITE(KOUT,12)' '
C  CALCULATE RKFSS
       CALL EQUATE(WM1,PHIM,IRFM,IRFM)
C  GMINV DESTROYS CALLING ARRAY
      MT=1
      CALL GMINV(IRFM,IRFM,WM1,WM2,MR,MT)
      CALL MAT1(WM2,BMD,IRFM,IRFM,ICBM,WM1)
      CALL MAT1(WM1,W,IRFM,ICBM,ICBM,WM2)
      CALL MSCALE(RKFSS,WM2,IRFM,IRHM,SQ)
      IO=5
      MSG='RKFSS, PICKED DIRECTLY IS'
      CALL MMATIO(RKFSS,IRFM,IRHM,IO,KIN,KOUT,NDIM,NDIM)
      GO TO 5
4     RETURN
```

```
      END
* DECK PRIMIT
      SUBROUTINE PRIMIT(WM2,U,ICBM,GCSTR,S,IRFM,BMD,WM1,PHIPRM,X,XPRIM,
     1 PHIM)
C   THIS SUBROUTINE COMPUTES THE PRIMED QUANTITIES NEEDED WHEN USING
C   KIEINMAN RICCATI SOLVER WITH NON ZERO CROSS COST WEIGHTING
C   MATRIX WXU
      REAL WM2(NDIM,NDIM),GCSTR(NDIM,NDIM),S(NDIM,NDIM),
     1BMD(NDIM,NDIM),WM1(NDIM,NDIM),PHIPRM(NDIM,NDIM),
     1 X(NDIM,NDIM),XPRIM(NDIM,NDIM),PHIM(NDIM,NDIM)
      REAL COM1(1),COM2(1)
      COMMON /MAIN1/ NDIM,NDIM1,COM1
      COMMON /MAIN2/COM2
      COMMON /INOU/KIN,KOUT,KPUNCH
C  NOW COMPUTE XPRIM,   PHIPRM
      CALL EQUATE(WM2,U,ICBM,ICBM)
C GMINV DESTRIYS THE CALLING ARRAY
      MT=1
      CALL GMINV(ICBM,ICBM,WM2,GCSTR,MR,MT)
C  GCSTR= UI ICBM  X  ICBM
      CALL MAT4(GCSTR,S,ICBM,ICBM,IRFM,WM2)
CWM2= UI(ST)    ICBM   X   IRFM
      CALL MAT1(BMD,WM2,IRFM,ICBM,IRFM,WM1)
      C1=-1.0
      CALL MADD1(IRFM,IRFM,PHIM,WM1,PHIPRM,C1)
C PHIPRM= PHIM-BMD(UI)ST    IRFM    X   IRFM
      CALL MAT1(S,WM2,IRFM,ICBM,IRFM,WM1)
      CALL MADD1(IRFM,IRFM,X,WM1,XPRIM,C1)
C  XPRIM= X-S(UI)ST    IRFM   X   IRFM
C
      RETURN
      END
* DECK RGS
      SUBROUTINE RGS(GCSTR,RKFSS,GCX,GCY,GCZ,BCY,BCZ,FC,YD,
     1 WM1,WM2,WM3,WM4,WM5,WM6,WM7,WM8,WM9,WM10,WV1,WV2,
     1 WMA,WMB,WMC,WMD,WME,WMF,FT,BT,GT,HT,QT,RT,
     1 FM,BM,GM,HM,QM,RM,XO,PO,WXX,WUU,WXU,FA,BA,GA,QA,GUA,
     1MA4,FXA,RA,PXVA,GCZA,IRY,IFLGCZ,IFLGSD,
     1MU,PUMAX,PUMIN,PXTMAX,PXTMIN,MUMAX,MUMIN,MXAMAX,MXAMIN,
     1 PUOUT,PXTOUT,MXTOUT,MUOUT,WV3,WV4)
      REAL COM1(1),COM2(1)
C
C  THIS SUBROUTINE SETS UP A CONTINUOUS, DISCRETIZED CONTINUOUS OF A
C   SAMPLED DATA LQG CONTROLLER BASED ON USERS REQUEST.  EACH
C   CONTROLLER IS PUT INTO THE PROPER FORMAT FOR  THE PERFORMANCE
C   ANALYSIS SUBROUTINE, PERFAL.

      CHARACTER MSG*60
      REAL FT (NDIM,NDIM),BT(NDIM,NDIM),GT(NDIM,NDIM),
     1 HT   (NDIM,NDIM),RM   (NDIM,NDIM),FM(NDIM,NDIM),
     1BM(NDIM,NDIM),GM   (NDIM,NDIM),XO(NDIM),
     1HM(NDIM,NDIM),QM   (NDIM,NDIM),PO(NDIM,NDIM),
     1QT(NDIM,NDIM),RT   (NDIM,NDIM),
     1GCSTR(NDIM,NDIM),RKFSS(NDIM,NDIM),
     1WV1(NDIM),WV2(NDIM),WXU(NDIM,NDIM),
     1WM1   (NDIM,NDIM),WM2  (NDIM,NDIM),WM3(NDIM,NDIM),
     1WM4   (NDIM,NDIM),WM5  (NDIM,NDIM),WM6(NDIM,NDIM),
     1WM7   (NDIM,NDIM),WM8  (NDIM,NDIM),WM9(NDIM,NDIM),
     1WM10  (NDIM,NDIM),WMA(NDIM2,NDIM2),WMB(NDIM2,NDIM2),
     1WMC(NDIM2,NDIM2),WMD(NDIM2,NDIM2),WME(NDIM2,NDIM2),
     1WMF(NDIM2,NDIM2),WUU(NDIM,NDIM),WXX(NDIM,NDIM),
     1FA(NDIM2,NDIM2),BA(NDIM2,NDIM2),GA(NDIM2,NDIM2),
     1MA4(NDIM2),GCX(NDIM,NDIM),GCZ(NDIM,NDIM),
     1QA(NDIM2,NDIM2),PXA(NDIM2,NDIM2),PXVA(NDIM2,NDIM2),
     1MUOUT(NDIM),MXTOUT(NDIM),PXTOUT(NDIM)
         REAL PUOUT(NDIM),YD(NDIM3),GUA(NDIM2,NDIM2),
     1MXAMIN(NDIM),MXAMAX(NDIM),MUMIN(NDIM),
     1MUMAX(NDIM),PXTMIN(NDIM),PXTMAX(NDIM),
     1PUMIN(NDIM),PUMAX(NDIM),GCZA(NDIM2,NDIM2),
     1RA(NDIM2,NDIM2),BCY (NDIM,NDIM),BCZ(NDIM,NDIM),
     1WV3(NDIM2),WV4(NDIM2),
     1GCY  (NDIM,NDIM),FC  (NDIM,NDIM)
      INTEGER IFLGCZ,IRY
      REAL MU(NDIM)
      COMMON /RNTIM/ RNTIME,DELTIM
      COMMON /MAIN2/COM2
      COMMON /MAIN1/NDIM,NDIM1,COM1
      COMMON / INOU/ KIN,KOUT,KPUNCH
      COMMON /MAIN4/NDIM2,NDIM3
      COMMON /MAUN5/ MSG
      COMMON /MAIN6/ ICBT,ICBM,ICFA,ICGA,ICGM,ICGT,ICQA,IRFA,IRFM,IRFT,
     1    IRHT,IRQA,IO,LOG,IRHM,NUPOTS
C
      WRITE(KOUT,*)'ENTER A C FOR CONTINUOUS TIME LQG CONTROLLER AND A
     1 D FOR A SAMPLED DATA LQG CONTROLLER>'
      READ(KIN,12,END=2933)MSG
      IF (MSG.EQ.'C') THEN
      IFLGSD=0
      CALL CLQGRS(GCSTR,FM,BM,RKFSS,HM,GCX,GCY,GCZ,BCY,BCZ,FC,YO,
     1  RM,QM,FT,BT,GT,CT,RT,HT,
     1IRY,IFLGCZ,WM1,WM2,WM3
     1 ,PO,GM,WM4,WM5,WM6,WV1,WV2,WUU,WXX,XO,WXU,WM7,WM8)
      IF (IO.EQ.0)THEN
      GO TO 2933
      END IF
      CALL FRMAUG(QT,RT,FT,BT,GCZ,HT,GCX,BCZ,FC,GCY,BCY,GT,XO,PO,FA,BA,
     1 GA,QA,QUA,WM1,WM2,WMA,WMB,WMC,WMD,WME,WMF,
     1 MXA,PXA,FA,PXVA,GCZA,IRY,IFLGCZ,IFLGSD)
      WRITE(KOUT,*)' '
      WRITE(KOUT,*)'DO YOU WISH TO CALCULATE THE EIGENVALUES OF THE CLOS
     1ED-LOOP F MATRIX?  Y   OR   N>'
      READ(KIN,12)MSG
      IF (MSG.EQ.'Y') THEN
      NSAV1=NDIM
      NDIM=NDIM2
      NDIM1=NDIM2+1
      WRITE(KOUT,*)'THE EIGENVALUES OF THE CLOSED-LOOP F MATRIX  ARE...'
      CALL MEIGN(FA,WV3,WV4,IRFA,WMA)
      NDIM=NSAV1
      NDIM1=NDIM+1
      END IF
      CALL MYINTG(WMA,WMB,WMC,WME,GA,QA,FA,BA,IRFA,ICGA,IRY,IRQA)
      ELSE
C  SOME SAMPLED DATA CONTROLLER IS WANTED
       IFLGSD=1
      WRITE(KOUT,*)'DO YOU WISH TO MERELY DISCRETIZE THE CONTINUOUS TIME
     1 CONTROLLER,   Y  OR    N>'
      READ(KIN,12,END=2933)MSG
      IF (MSG.EQ.'Y')THEN
      WRITE(KOUT,*)'********NOTE THAT WHEN ENTERING THE TIME INCREMENT
     1 IN THE CONTINUOUS TIME CONTROLLER SET UP,   REMEMBER IT WILL BECO
     1ME THE SAMPLE TIME FOR THE DISCRETIZED CONTROLLER*********'
      CALL CLQGRS(GCSTR,FM,BM,RKFSS,HM,GCX,GCY,GCZ,BCY,BCZ,FC,YO,
     1  RM,QM,FT,BT,GT,CT,RT,HT,IRY,IFLGCZ,WM1,WM2,WM3,PO,GM,
     1  WM4,WM5,WM6,WV1,WV2,WUU,WXX,XO,WXU,WM7,WM8)
       IF (IO.EQ.0)THEN
       GO TO 2933
       END IF
       CALL DSCRTZ(WM1,WM2,WM3,WM4,IRFM,DELTIM,FC,BCY,IRY,BCZ,IRHM,
     1 WM5,WM6,GCSTR,GT,QT,FT,BT,IRFT,ICGT,ICBT,IRHT,RKFSS,RT,GCX,
     1 ICBM)
       WRITE(KOUT,*)' '
      WRITE(KOUT,*)'DO YOU WANT TO CALCULATE THE EIGENVALUES OF THE STAT
     1E TRANSITION MATRIX FOR THE DISCRETIZED CONTROLLER?  Y  OR  N>'
      READ(KIN,12)MSG
      IF (MSG.EQ.'Y') THEN
      WRITE(KOUT,*)'THE EIGENVALUES OF THE STATE TRANSITION MATRIX FOR T
     1HE DISCRETIZED CONTROLLER ARE...'
      CALL MEIGN(WM2,WV1,WV2,IRFM,BCY)
      END IF
C  RKFSS, IN PRECEDING CALL STATEMENT ARE MERELY DUMMY WORK SPACES
C  GCSTR CONTAINS BCZD UPON RETURN FROM DSCRTZ
      CALL FRMAUG(WM6,RKFSS,WM5,GCSTR,GCZ,HT,GCX,WM4,WM2,GCY,WM3,WM1,XO,
     1 PO,FA,BA,GA,CA,GUA,BCY,FC,WMA,WMB,WMC,WMD,WME,WMF,MXA,PXA,RA,
     1 PXVA,GCZA,IRY,IFLGCZ,IFLGSC)
      ELSE
      IFLGCZ=0
      CALL DLQGRS(GCX,GCY,GCZ,BCY,BCZ,WM3,FC,WM2,WM5,WM1,WM4,FM,BM,
     1 QM,GM,RM,HM,GT,QT,FT,BT,RT,HT,WXX,WUU,GCSTR,RKFSS,YO,
     1 IRY,IFLGCZ,WXU,WM6,WM7,WM8,WM9,WM10,WV1,WV2)
      CALL FRMAUG(WM1,WM2,WM3,WM4,GCZ,HT,GCX,BCZ,FC,GCY,BCY,WM5,XO,
     1 PO,FA,BA,GA,CA,GUA,RKFSS,GCSTR,WMA,WMB,WMC,WMD,WME,WMF,MXA,PXA,
     1 RA,PXVA,GCZA,IRY,IFLGCZ,IFLGSD)
C  RKFSS,GCSTR ARE DUMMY WORK SPACE IN CALL TO FRMAUG
      END IF
      NSAV1=NDIM
      NSAV2=NDIM1
      NDIM1=NDIM2+1
      NDIM=NDIM2
      CALL EQUATE(WMA,FA,IRFA,IRFA)
      CALL EQUATE(WMC,BA,IRFA,IRY)
       CALL MAT1(GA,QA,IRFA,IRQA,IRQA,WME)
      CALL MAT4(WME,GA,IRFA,IRQA,IRFA,WMB)
      NDIM=NSAV1
      NDIM1=NSAV2
C  NOTE FC AND BCY IN CALL TO FRMAUG ARE DUMMY WORKSPACES
      END IF
      RETURN
2933  IO=0
   12    FORMAT(A1)
      END
* DECK XSU
      SUBROUTINE XSU(X,S,U,PHIJO,BJO,PHII,INTPII,PHIJ,BJ,
     1 TEMP,TEMP1,TEMP2,FM,BM,IRFM,ICBM,WXX,WUU,WXU,PHIT)
C   THIS ROUTINE COMPUTES X S  AND U AT TIME T-SUB-I FOR USE IN THE
C   SAMPLED DATA CONTROLLERDETERMINISTIC GAIN CALCULATION.
C   THIS ROUTINE APPROXIMATES THE INTEGRALSREQUIRED (SEE MAYBECK,
C   EQUATIONS 14.25) BY TREATING  TIME VAYING ENTITIES IN  THE
C   INTEGRANDS AS CONSTANTS OVER SOME SUBINTERVAL OF THE SAMPLE TIME
C   THAT IS CHOSEN BY THE USER
      REAL X   (NDIM,NDIM), S(NDIM,NDIM), U(NDIM,NDIM),
     1PHIJO(NDIM,NDIM),BJO(NDIM,NDIM),PHII(NDIM,NDIM),
     1INTPII  (NDIM,NDIM),PHIJ(NDIM,NDIM),BJ(NDIM,NDIM),
     1TEMP (NDIM,NDIM),TEMP1(NDIM,NDIM),TEMP2(NDIM,NDIM),
     1FM(NDIM,NDIM),BM  (NDIM,NDIM),WXX(NDIM,NDIM),
     1WUU  (NDIM,NDIM),WXU(NDIM,NDIM),PHIT(NDIM,NDIM)
      REAL COM1(1),COM2(1)
      CHARACTER MSG*60
       COMMON /MAIN1/NDIM,NDIM1,COM1
      COMMON/MAIN2/COM2
      COMMON /INOU/ KIN,KOUT,KPUNCH
       COMMON /RNTIM/ RNTIME,DELTIM
      COMMON /MAUNB/ MSG
      IZIT=0
      DO 782 IJK=1,1000
C  GIVE USER UP TO 1000 CHANCES TO CHOOSE DIFFERENT SUBINTERVAL LENGTH
      IF(IZIT.EQ.1)THEN
      WRITE(KOUT,12)' '
      WRITE(KOUT,*)'DO YOU WISH TO RECOMPUTE X,S,U, BASED ON A DIFFERENT
     1 SUBINTERVAL LENGTH,  Y  OR  N>'
      READ(KIN,11)MSG
11    FORMAT(A1)
12    FORMAT(A1,/)
      IF (MSG.EQ.'N')THEN
      RETURN
      END IF
      END IF
      IZIT=1
      WRITE(KOUT,*)'ENTER THE NUMBER OF SUBINTERVALS TO USE IN THE APPRO
     1XIMATIONS  OF INTEGRALS NEEDED TO CALCULATE X, S, AND U (SUGGEST 5
     1 OR MORE )>'
      READ(KIN,*)INTVAL
C  NOW INITIALIZE VARIABLES REQUIRED IN CALCULATIONS
      C1=1.0
      CALL IONT(IRFM,PHIJO,C1)
      C1=0.0
       CALL IONT(IRFM,X,C1)
      CALL IONT(ICBM,U,C1)
        DO 13 I=1,IRFM
       DO 13 J=1,ICBM
      BJO(I,J)=0
13     S(I,J)=0
C  INITIALIZATION COMPLETE,   NOW COMPUTE PHIJ, INTPHJ FOR SUBINTERVAL
C  PHI,INTPHI ARE APPROXIMATED BY TAKING AVERAGE OF VALUES AT THE
C BEGINNING ,END AND 8 POINTS IN THE MIDDLE OF EACH SUBINTERVAL
C   THIS MEANS 9 SUB SUB INTERVAL PCINTS TO BE CALCULATED.  HOWEVER, O
C NLY 1 CALL TO INTEGRATE ROUTINIS REQUIRED SINCE FM IS A CONSTANT
C MATRIX
      DEL=0.0
      SUBINT=DELTIM/(4*INTVAL)
      DO 23 JK=1,INTVAL
C  COMPUTE PHIJ BJ AND THEN UPDATE PHIJC,BJO
C    FCR EACH SUBINTERVAL
      C1=1.0
      DO 371 INTVL=1,4
      DEL=DEL+SUBINT
      CALL DSCRT(IRFM,FM,DEL,PHII,INTPII,5)
      CALL MADD1(IRFM,IRFM,PHIJC,PHII,TEMP,C1)
      CALL EQUATE(PHIJO,TEMP,IRFM,IRFM)
C PHIJC=PHIJO+PHII
      CALL MAT1(INTPII,BM,IRFM,IRFM,ICBM,TEMP1)
      CALL MADD1(IRFM,ICBM,BJO,TEMP1,TEMP,C1)
      CALL EQUATE(BJO,TEMP,IRFM,ICBM)
C  BJO=BJO+INTPII*BM
371   CONTINUE
C NOW CALCULATE BJ PHIJ
      C1=0.2
      CALL MSCALE(PHIJ,PHIJO,IRFM,IRFM,C1)
      CALL MSCALE(BJ,BJO,IRFM,ICBM,C1)
C  BJ,  PHIJ   NOW AVAILABLE FOR THIS SUBINTERVAL
C  RESET PHIJO,BJO
      CALL EQUATE(PHIJO,PHII,IRFM,IRFM)
      CALL EQUATE(BJO,TEMP1,IRFM,ICBM)
C  NOW UPDATE X,S,U FOR THIS SUBINTERVAL
C  X=SUM OF (PHIJT*WXX*PHIJ*(DELTIM/INTVAL)) FOR ALL JK
C  S=SUM OF ((PHIJT*WXX*BJ+PHIJT*WXU)*(DELTIM/INTVAL))
      CALL MAT4A(PHIJ,WXX,IRFM,IRFM,IRFM,TEMP)
C TEMP= PHIJT * WXX
      CALL MAT1(TEMP,PHIJ,IRFM,IRFM,IRFM,TEMP1)
      CALL MAT1(TEMP,BJ,IRFM,IRFM,ICBM,TEMP2)
      C1A=DELTIM/INTVAL
      CALL MADD1(IRFM,IRFM,X,TEMP1,TEMP,C1A)
      CALL EQUATE(X,TEMP,IRFM,IRFM)
C  X IS NOW UPDATED FOR THIS SUB-SUB INTERVAL
```

```
      CALL MAT4A(PHIJ,WXU,IRFM,IRFM,ICBM,TEMP1)
      C1=1.0
      CALL MADD1(IRFM,ICBM,TEMP2,TEMP1,TEMP,C1)
      CALL MADD1(IRFM,ICBM,S,TEMP,TEMP1,C1A)
      CALL EQUATE(S,TEMP1,IRFM,ICBM)
C   S UPDATE FOR THIS SUB-SUB INTERVAL NOW COMPLETE
C
C  U=SUM OF((BJT*WXX*BJ+WUU+BJT*WXU+WXUT*BJ)*(DELTIM/INTVAL)) FOR ALL JK
      C1=1.0
      CALL MAT3A(ICBM,IRFM,BJ,WXX,TEMP)
      CALL MADD1(ICBM,ICBM,TEMP,WUU,TEMP1,C1)
      CALL MAT4A(BJ,WXU,ICBM,IRFM,ICBM,TEMP)
      CALL MADD1(ICBM,ICBM,TEMP1,TEMP,TEMP2,C1)
      CALL MAT4A(WXU,BJ,ICBM,IRFM,ICBM,TEMP)
      CALL MADD1(ICBM,ICBM,TEMP2,TEMP,TEMP1,C1)
      CALL MADD1(ICBM,ICBM,U,TEMP1,TEMP,C1A)
      CALL EQUATE(U,TEMP,ICBM,ICBM)
C   U   UPDATB FOR THIS SUB-SUB INTERVAL NOw COMPLETE
 23      CONTINUE
      IO=5
      MSG='X(TI) IS'
      CALL MMATIO(X,IRFM,IRFM,IO,KIN,KOUT,NDIM,NDIM)
      MSG='U(TI) IS '
      CALL MMATIO(U,ICBM,ICBM,IO,KIN,KOUT,NDIM,NDIM)
      MSG='S(TI) IS'
      CALL MMATIO(S,IRFM,ICBM,IO,KIN,KOUT,NDIM,NDIM)
 782  CONTINUE
      RETURN
      END
      SUBROUTINE CNOISE(FM,GM,HM,BM,QM,IRFM,IRHM,ICGM,ICBM,WM1,WM2,WM3)
      DIMENSION FM(NDIM,NDIM),HM(NDIM,NDIM),BM(NDIM,NDIM),QM(NDIM,NDIM),
     1GM(NDIM,NDIM),COM1(1),WM1(NDIM,NDIM),WM2(NDIM,NDIM),WM3(NDIM,NDIM)
      COMMON/MAIN1/NDIM,NDIM1,COM1
      COMMON/INCU/KIN,KOUT,KPUNCH
 25   FORMAT(I1)
      WRITE(KOUT,*)'1ST OR 2ND ORDER SHAPING FILTER (1 OR 2)>'
      READ(KIN,25)ISIZE
C   CALCULATE THE DIMENSION OF THE AUGMENTED SYSTEM
      IRFS=IRFM+(ISIZE*ICBM)
      IF(ISIZE.NE.1)GO TO 10
      CALL FORDER(FM,BM,GM,QM,HM,IRFM,IRHM,ICBM,ICGM,IRFS,WM1)
      GO TO 20
 10   CALL SORDER(FM,BM,GM,QM,HM,IRFM,IRHM,ICBM,ICGM,IRFS,WM1,WM2,WM3)
 20   CONTINUE
      RETURN
      END
      SUBROUTINE FORDER(FM,BM,GM,QM,HM,IRFM,IRHM,ICBM,ICGM,IRFS,WU)           A
      DIMENSION FM(NDIM,NDIM),BM(NDIM,NDIM),GM(NDIM,NDIM),QM(NDIM,NDIM),
     1HM(NDIM,NDIM),COM1(1),COM2(1),WU(NDIM,NDIM)
      COMMON/MAIN1/NDIM,NDIM1,COM1
      COMMON/MAIN2/COM2
      COMMON/INCU/KIN,KOUT,KPUNCH
      COMMON/MAUN5/MSG
      REAL ENTRY,ENTRY1
      CHARACTER MSG*60
C   ZERO OUT BOTTOM PARTITIONS OF GM,BM,GM
      IFS=IRFM+1
      IGS=ICGM+1
      ICGS=ICGM+ICBM
      CALL ZPART(FM(IFS,1),ICBM,IRFS)
      CALL ZPART(BM(IFS,1),ICBM,ICBM)
      CALL ZPART(GM(IFS,1),ICBM,ICGS)
      CALL ZPART(HM(1,IFS),IRHM,ICBM)
      CALL ZPART(QM(IGS,1),ICBM,ICGS)
      CALL ZPART(QM(1,IGS),ICGM,ICBM)
      CALL ZPART(HU,ICBM,ICBM)
```

```
C   DESIGN ONE SHAPING FILTER FOR EACH CONTROL CHANNEL
      DO 80 IF=1,ICBM
      WRITE(KOUT,*)'FILTER DESIGN ',IF
      WRITE(KOUT,*)'INPUT OPTIONS:  (1) ENTER A AND B,
     1(2) ENTER Q, (3) GO TO NEXT FILTER OR EXIT IF DONE'
      DO 90 IT=1,50
27    FORMAT(I1)
      WRITE(KOUT,*)'OPTION>'
      READ(KIN,27)IOPT
      GO TO (1,2,3)IOPT
1     WRITE(KOUT,*)'ENTER A AND B>'
      READ(KIN,*)ENTRY,ENTRY1
      HU(IF,IF)=ENTRY-ENTRY1
      FM((IRFM+IF),(IRFM+IF))=-1*ENTRY1
      GO TO 90
2     WRITE(KOUT,*)'ENTER Q>'
      READ(KIN,*)ENTRY
      QM((ICGM+IF),(ICGM+IF))=ENTRY
      GO TO 90
90    CONTINUE
3     CONTINUE
80    CONTINUE
C   CALCULATE BM*HU AND PUT IN UPPER RIGHT PARTITION OF FM
      CALL MAT1(BM,HU,IRFM,ICBM,ICBM,FM(1,IFS))
C   INSERT BM INTO UPPER RIGHT PARTITION OF GM, DU=I
      CALL EQUATE(GM(1,IGS),BM,IRFM,ICBM)
C   PUT AN IDENTITY MATRIX IN LOWER RIGHT PARTITION OF GM
      CALL IDNT(ICBM,GM(IGS,IGS),1.)
C   CHANGE COLUMN DIMENSION OF GM AND ORDER OF SYSTEM
      ICGM=ICGS
      IRFM=IRFS                                                     A
      IHOLD=IO
      IO=5
      MSG='QMAUG'
      CALL MMATIO(QM,ICGM,ICGM,IO,KIN,KOUT,NDIM,NDIM)
      IO=IHOLD
      RETURN
      END
      SUBROUTINE SORDER(FM,BM,GM,QM,HM,IRFM,IRHM,ICBM,ICGM,IRFS,
     1HU,FU,GU)
      DIMENSION FM(NDIM,NDIM),BM(NDIM,NDIM),GM(NDIM,NDIM),
     1QM(NDIM,NDIM),HM(NDIM,NDIM),HU(NDIM,NDIM),COM1(1),
     1COM2(1),FU(NDIM,NDIM),GU(NDIM,NDIM)
      COMMON/MAIN1/NDIM,NDIM1,COM1
      COMMON/MAIN2/COM2
      COMMON/INOU/KIN,KOUT,KPUNCH
      COMMON/MAUN5/MSG
      REAL ENTRY,ENTRY1,BAC,BTC
      IFS=IRFM+1
      IGS=ICGM+1
      ICGS=ICGM+ICBM
      ICBM2=2*ICBM
      CALL ZPART(FM(IFS,1),ICBM2,IRFS)
      CALL ZPART(BM(IFS,1),ICBM2,ICBM)
      CALL ZPART(QM(IGS,1),ICBM,ICGS)
      CALL ZPART(QM(1,IGS),ICGM,ICBM)
      CALL ZPART(GM(IFS,1),ICBM2,ICGS)
      CALL ZPART(GM(1,IGS),IRFM,ICBM)
      CALL ZPART(HM(1,IFS),IRHM,ICBM2)
      CALL ZPART(FU,ICBM2,ICBM2)
      CALL ZPART(HU,ICBM,ICBM2)
      CALL ZPART(GU,ICBM2,ICBM)
C   DESIGN ONE SHAPING FILTER PER CONTROL CHANNEL
      DO 80 IF=1,ICBM
      WRITE(KOUT,*)'FILTER DESIGN',IF
      WRITE(KOUT,*)'INPUT OPTIONS: (1) ENTER C, (2) ENTER D AND E,
     1(3) ENTER Q, (4) GO TO NEXT FILTER OR EXIT IF DONE'
      DO 80 IF=1,FC
      WRITE(KOUT,*)'OPTION>'
 26   FORMAT(I1)
      READ(KIN,26)IOPT
      GO TO (1,2,3,4)IOPT
 1    WRITE(KOUT,*)'ENTER A>'
      READ(KIN,*)ENTRY
      HU(IF,(2*IF-1))=ENTRY
      HU(IF,(2*IF))=1.
      GO TO 90
 2    WRITE(KOUT,*)'ENTER B AND C>'
      READ(KIN,*)ENTRY,ENTRY1
      BAC=-1*(ENTRY+ENTRY1)
      BTC=-1*(ENTRY*ENTRY1)
      FU((2*IF-1),(2*IF))=1.
      FU((2*IF),(2*IF-1))=BTC                                           A
      FU((2*IF),(2*IF))=BAC
      GO TO 90
 3    WRITE(KOUT,*)'ENTER Q>'
      READ(KIN,*)ENTRY
      QM((ICGM+IF),(ICGM+IF))=ENTRY
 90   CONTINUE
 4    CONTINUE
      GU((IF*2),IF)=1.
 80   CONTINUE
C   CALCULATE BM*HU AND PUT IN UPPER RIGHT PARTITION OF FM
      CALL MMUL(BM,HU,IRFM,ICBM,ICBM2,FM(1,IFS))
C   INSERT FU INTO LOWER RIGHT PARTITION OF FM
      CALL EQUATE(FM(IFS,IFS),FU,ICBM2,ICBM2)
C   INSERT GU INTO LOWER RIGHT PARTITION OF GM
      CALL EQUATE(GM(IFS,IGS),GU,ICBM2,ICBM)
C   CHANGE COLUMN DIMENSION OF GM AND ORDER OF SYSTEM
      ICGM=ICGS
      IRFM=IRFS
      RETURN
      END
```

APPENDIX C: Modifications and Additions to CGTPI

C.1 Introduction

This appendix is intended to be used as a supplement to References 13 and 30. These sequences are user's guides to a computer-aided design program for designing controllers employing a Command Generator Tracker in the feedforward loop and a Proportional-plus-Integral regulator in the feedback loop (CGTPI). Reference 13 describes the original version of the program. Reference 30 contains a modified version that admits implicit model following for design of controller gains and that can be run in conjunction with a performance evaluation program (PFEVAL), described in the same reference. Modifications outlined in this chapter are made to the version in Reference 30.

As stated in Chapter III, only the PI regulator design portion of the program is used for this thesis. However, to execute PFEVAL, the user must first execute all three design paths of CGTPI. These include the PI regulator design, CGT design, and Kalman filter design. By proper choice of the CGT model matrices, the controlled degenerates into a simple PI regulator. These matrices are defined below:

$$A_m = [I] \tag{C-1}$$

$$B_m = [0] \tag{C-2}$$

$$C_m = [0] \tag{C-3}$$

$$D_m = [I] \tag{C-4}$$

Modifications to the program include an additional subroutine to apply the Doyle and Stein technique. This is bracketed and marked with an "A" in the source code at the end of this appendix. Modifications to allow inputting colored-noise into the design model are bracketed and marked with an "M". Since the modifications made to CGTPI are not extensive, only the specific subroutines that contain changes are listed.

C.2 Additions to CGTPI

One additional subroutine is added to CGTPI to allow modification of the dynamic driving noise covariance matrix, Q_d, via the Doyle and Stein technique as described in Section 3.4 Subroutine DAS is listed in lines 29730 through 29940. When executing the Kalman filter design option of CGTPI, this subroutine is called in line 18730. An input prompt will ask if the Doyle and Stein modification is desired. If so, then a prompt will ask for the scalar design parameter, q. The Q_d matrix is then modified as expressed in Equation (3-43) with the assumption that V = I. The modified Q_d matrix is also written to the LIST file.

C.3 Modifications to CGTPI

The original version of CGTPI was not written with the intention of augmenting additional states with the design model. The modifications listed in the source code in this appendix will allow the technique of injecting time-correlated noise into the design model, but the method is awkward as will become apparent.

To apply the technique requires two executions of CGTPI. In the first execution, the unaugmented design model and truth model are entered to the program. Then, the PI regulator design, CGT design, and Kalman filter design paths are executed normally. All prompts to enter either filter gains or CGT gains should be answered with an "N". In the CGT design path, the model matrices listed in Equations (D-1) through (D-4) can be entered if only a PI regulator is desired. It is useful to write the truth model to the SAVE file. An input prompt asks if it is desired to write the model to the SAVE file. If a "Y" is entered, the model matrices are stored by the program. This file is structured so that the design model, truth model, regulator gains, CGT gains, and command model can all be written to the file for later use. Copying the SAVE file to a DATA file allows the model and gain matrices to be read directly into the program, rather than be inputted from a terminal (Ref 13). The regulator and CGT gains are also written to the SAVE file. After following all three design paths, execution of the program is halted.

Prior to the second execution, the SAVE file is copied to a DATA file. Also, it is necessary to have the augmented matrices specified in Chapter IV available to be read into the program from the terminal. In the second execution, the augmented system is entered into the design model. The truth model can be read from the DATA file containing information from the previous execution. After entering the model, the CGT path is followed again. A prompt will ask if it is desired to read in the CGT gains. A "Y" is entered, the gains are read from the DATA file, and this path is exited. This modification is listed in lines 6600 to 6630 and 6910.

Next, the Kalman filter design path is followed with the augmented design model. After execution of this option, a prompt will inquire if it is desired to store performance analysis data to the SAVE file. A "Y" is entered, and another prompt will inquire if it is desired to read controller gains from the DATA file. Again, a "Y" is entered. The modifications in lines 27230 through 27430 allow the program to read in PI controller gains from the previous execution and augment with columns by zeros so that the shaping filter states are not fed back through the controller. Performance evaluation data is stored and PFEVAL can thus be run normally as described in Reference 30.

The modified version also allows the option of reading in Kalman filter gains from a previous execution stored on the DATA file. These modifications are listed in lines 4230 to 4320.

Final modifications are given in lines 4470 through 4610 to reformat the PI controller gains and allow storage room for the augmented zeros.

```
      SUBROUTINE CGTXQ
      COMMON/MAIN1/NDIM,NDIM1,COM1(1)
      COMMON/MAIN2/COM2(1)
      COMMON/INOU/KIN,KOUT,KPUNCH
      COMMON/DESIGN/NVCOM,TSAMP,LFLRPI,LFLCGT,LFLKF,LTEVAL,LABORT
      COMMON/FILES/KSAVE,KDATA,KPLOT,KLIST,KTERM
      COMMON/SYSMTX/NVSM,SM(1)
      COMMON/ZMTX1/NVZM,ZM1(1)
      COMMON/ZMTX2/ZM2(1)
      COMMON/NDIMD/NND,NRD,NPD,NMD,NDD,NWD,NWDD,NPLD,NWPNWD,NNPR
      COMMON/LOCD/LAP,LGP,LPHI,LBD,LEX,LPHD,LQ,LQN,LQD,LC,LDY,LEY
     1,LHP,LR
      COMMON/DSNMTX/NVDM,NODY,NOEY,DM(1)
      COMMON/NDIMC/NNC,NRC,NPC
      COMMON/LOCC/LPHC,LBDC,LCC,LDC
      COMMON/CMDMTX/NVCM,NEWCM,NODC,CM(1)
      COMMON/NDIMT/NNT,NRT,NMT,NWT
      COMMON/LOCT/LPHT,LBDT,LQDT,LHT,LRT,LTDT,LTNT
      COMMON/TRUMTX/NVTM,TM(1)
      COMMON/LCNTRL/LPI11,LPI12,LPI21,LPI22,LPHDL,LBDL
      COMMON/CONTROL/NVCTL,CTL(1)
      COMMON/LREGPI/LXDW,LUDW,LPHCL,LKX,LKZ
      COMMON/CREGPI/NVRPI,RPI(1)
      COMMON/LCGT/LA11,LA13,LA21,LA23,LA12,LA22,LKXA11,LKXA12,
     1LKXA13
      COMMON/CCGT/NVCGT,CGT(1)
      COMMON/LKF/LEADSN,LFLTRK,LFCOV
      COMMON/CKF/NVFLT,FLT(1)
      COMMON/AMC/AM(1)
      COMMON/BDG/BD(1)
      DIMENSION LD(15),ND(15)
      DATA NPLTZM/606/
      DATA IEOI,NO/-1,1HN/
      REWIND KLIST
      WRITE(KLIST,115) DATE(DUM),TIME(DUM)
      WRITE(KTERM,115) DATE(DUM),TIME(DUM)
  115 FORMAT('1',27X,'* * * CGTPIF * * *'/14X,
     1 'PROGRAM TO DESIGN A COMMAND GENERATOR TRACKER'/8X,
     2 'USING A REGULATOR WITH PROPORTIONAL PLUS INTEGRAL CONTROL
     1'/16X,
     3 'AND A KALMAN FILTER FOR STATE ESTIMATION.'/28X,
     4 '* * * CGTPIF * * *'//11X,'DATE : ',A10//,11X,
     5 'TIME : ',A10////)
      REWIND KSAVE
      REWIND KDATA
      WRITE(KSAVE,112) IEOI,NPLTZM
      DO 10 I=1,15
   10 ND(I)=0
      DO 12 I=1,15
   12 LD(I)=1
      LFLRPI=0
      LFLCGT=0
      LFLKF=0
      LTEVAL=0
      LABORT=0
      IPT=0
      ICGT=0
      ITRU=0
      IFLTR=0
      ICODE=4
      LFAVAL=0
      LGCGT=0
      NVCOM=MINO(NDIM,NVZM)
      KOUT=KLIST
      KPUNCH=KPLOT
      IF(NVSM.GE.NPLTZM) GO TO 50
      WRITE 101,NPLTZM
      GO TO 1000
 50   WRITE 102
      READ*,TSAMP
      IF(TSAMP.LE.0.) GO TO 50
      WRITE(KLIST,103) TSAMP
 103  FORMAT('OSAMPLE PERIOD IS ',F3.3,' SECONDS')
      CALL SETUP(ND,LD,ICGT,ITRU,1)
      IF(LABORT) 1000,100,1000
 100  LABORT=0
      WRITE 104
 104  FORMAT('OCONTROLLER DESIGN (Y OR N) >')
      READ 111,IANS
      IF(IANS.EQ.NO) GO TO 500
      LFLKF=0
      CALL PIMTX(IPI)
      IF(LABORT) 1000,125,1000
 125  WRITE 105
 105  FORMAT('ODESIGN REG/PI (Y OR N) >')
      READ 111,IANS
      IF(IANS.EQ.NO) GO TO 150
      CALL SREGPI
      IF(LABORT) 1000,200,1000
 150  WRITE 106
 106  FORMAT('ODESIGN CGT (Y OR N) >')
      READ 111,IANS
      IF(IANS.EQ.NO) GO TO 100
      CALL SETUP(ND,LD,ICGT,ITRU,2)
      WRITE 117
117   FORMAT('OTERMINATE CGT DESIGN PATH (Y OR N)? >')
      READ 111,IANS
      IF(IANS.NE.NO) GO TO 500
      IF(ICGT) 155,100,155
 155  IF(LABORT) 100,160,1000
 160  CALL SCGT
      IF(LABORT) 100,170,1000
 170  IF(LFLCGT.LE.0) GO TO 125
 200  LABORT=0
      WRITE 107
 107  FORMAT('OCONTROLLER EVALUATION WRT TRUTH MODEL (Y OR N) >')
      READ 111,IANS
      IF(IANS.EQ.NO) GO TO 250
      CALL SETUP(ND,LD,ICGT,ITRU,3)
      IF(LABORT) 200,260,1000
 250  LTEVAL=0
 260  CALL CEVAL
      IF(LFLCGT.EQ.1) LGCGT=1
      IF(LFAVAL.EQ.0.OR.LGCGT.EQ.0) GO TO 100
 270  WRITE 600
 600  FORMAT('0WRITE PERFORMANCE EVALUATION DATA TO 'SAVE' FILE
     1(Y OR N)
     +>')
      READ 111,IANS
      IF(IANS.EQ.NO) GO TO 100
      ICODE=ICODE+1
      CALL PFDATA(ICODE,ND)
      INUM=ICODE-4
      WRITE 605,INUM
 605  FORMAT('0PERFORMANCE EVALUATION DATA, NO. 'I2,',WRITTEN TO
'SAVE
     +' FILE')
      GO TO 100
 500  LABORT=0
      WRITE 108
 108  FORMAT('0FILTER DESIGN (Y OR N) >')
      READ 111,IANS
      IF(IANS.EQ.NO) GO TO 900
      WRITE 777
777   FORMAT('0READ IN CGT GAINS (Y OR N) >')
      READ 111,IANS
      IF(IANS.EQ.NO) GO TO 505
M     IF (LGCGT.NE.0) GO TO 505
      IF (IFLTR.NE.0) GO TO 505
      CALL READFS(SM,ND,4,IERR)
      LGCGT=ND(2)
505   CALL FLTRK(IFLTR)
      IF(IFLTR.EQ.0) GO TO 900
      IF(LABORT) 1000,510,1000
 510  CALL SETUP(ND,LD,ICGT,ITRU,3)
      IF(LABORT) 500,525,1000
 525  CALL FEVAL
 530   IF(LABORT) 1000,540,1000
 540   LFAVAL=1
      IF(LGCGT.EQ.1) GO TO 270
      GO TO 500
 900  WRITE 109
 109  FORMAT('0END DESIGN RUNS (Y OR N) >')
      READ 111,IANS
      IF(IANS.EQ.NO) GO TO 100
      IF(LFLRPI.EQ.0) GO TO 1000
      NPNTS=NRD*NNPR
      ND(1)=NPNTS
      ND(2)=LGCGT
      ND(3)=LKX
      ND(4)=LKZ
      CALL FTMTX(RPI(LKX),SM,NPNTS,1)
M     ND(5)=NPNTS+1
      IF(LGCGT.EQ.0) GO TO 910
      ND(6)=LKXA11
      ND(7)=LKXA12
      ND(8)=LKXA13
      NPNTS=NRD*(NNC+NRC+NDD)
      ND(1)=ND(1)+NPNTS
      CALL FTMTX(CGT(LKXA11),SM(ND(5)),NPNTS,1)
910   CALL WFILED(4,ND(1),ND,SM)
      WRITE 113
 1000 CONTINUE
      WRITE(KLIST,110)
```

```
        REWIND KSAVE
        REWIND KDATA
        REWIND KLIST
        WRITE 110
  101   FORMAT('0INSUFFICIENT MEMORY /SYSMTX/, NEED: ',I4)
  102   FORMAT('0ENTER SAMPLE PERIOD FOR DIGITAL CONTROLLER >')
  110   FORMAT('0PROGRAM EXECUTION STOP')
  111   FORMAT(A3)
  112   FORMAT(2I4)
  113   FORMAT(6X,'REG/PI GAINS WRITTEN TO 'SAVE' FILE')
        RETURN
C END    SUBROUTINE CGTXQ
```

```
      SUBROUTINE SCMD(ND,LD,ICGT)
      COMMON/DESIGN/NVCOM,TSAMP,LFLRPI,LFLCGT,LFLKF,LTEVAL,LABORT
      COMMON/FILES/KSAVE,KDATA,KPLOT,KLIST,KTERM
      COMMON/SYSMTX/NVSM,SM(1)
      COMMON/ZMTX1/NVZM,ZM1(1)
      COMMON/ZMTX2/ZM2(1)
      COMMON/NDIMD/NND,NRD,NPD,NMD,NDD,NWD,NWDD,NPLD,NWPNWD,NNPR
      COMMON/NDIMC/NNC,NRC,NPC
      COMMON/CMDMTX/NVCM,NEWCM,NODC,CM(1)
      COMMON/LREGPI/LXDW,LUDW,LPHCL,LKX,LKZ
      COMMON/CREGPI/NVRPI,RPI(1)
      DIMENSION ND(1),LD(1)
      DATA NO/1HN/
      WRITE(KLIST,110)
110   FORMAT(////11X,5('* '),'CGT DESIGN',5(' *')////)
      NEWCM=0
      IF(LFLRPI) 10,5,10
5     WRITE 102
      READ 111,IANS
      IF(IANS.EQ.NO) GO TO 8
      CALL READFS(SM,ND,4,IERR)
      NSIZE=ND(5)-1
M     LKX=ND(3)
      LKZ=ND(4)
      CALL FTMTX(SM,RPI(LKX),NSIZE,1)
      IF(IERR.NE.0) RETURN
      CALL MATLST(RPI(LKX),NRD,NND,'KX',KLIST)
      CALL MATLST(RPI(LKZ),NRD,NRD,'KZ',KLIST)
      LFLRPI=-1
      GO TO 10
8     IF(LFLCGT.GE.0) GO TO 9
      WRITE 103
103   FORMAT('OSYSTEM UNSTABLE - - OPEN-LOOP CGT NOT FEASIBLE')
      RETURN
9     LKX=1
      LKZ=1
      NSIZE=NRD*NND
      CALL ZPART(RPI(LKX),1,NSIZE,1)
10    IF(ICGT.EQ.0) GO TO 12
      WRITE 108
108   FORMAT(' MODIFY COMMAND MODEL (Y OR N) >')
      READ 111,IANS
      IF(IANS.EQ.NO) RETURN
12    CALL RSYS(SM,LD,ND,2,ICGT)
      IF(LABORT.NE.0) RETURN
      NEWCM=1
      CALL POLES(SM,NNC,2,ZM1,ZM2)
      IF(NPC.EQ.NPD) GO TO 15
      WRITE 104
      LABORT=-1
      RETURN
15    CALL OSCRTC(LD,ZM1)
```

M

```
 102  FORMAT(' READ REG/PI GAINS FROM 'DATA' FILE (Y OR N) >')
 104  FORMAT('OCOMMAND AND DESIGN MODEL OUTPUTS NOT EQUAL IN
     1NUMBER')
      RETURN
C END SUBROUTINE SCMD
```

```
  18250=        SUBROUTINE FLTRK(IFLTR)
  18260=        COMMON/MAIN1/NDIM,NDIM1,COM1(1)
  18270=        COMMON/MAIN2/COM2(1)
  18280=        COMMON/DESIGN/NVCOM,TSAMP,LFLRPI,LFLCGT.LFLKF,LTEVAL.LABORT
  18290=        COMMON/FILES/KSAVE,KDATA.KPLOT,KLIST.KTERM
  18300=        COMMON/SYSMTX/NVSM,SM(1)
  18310=        COMMON/ZMTX1/NVZM,ZM1(1)
  18320=        COMMON/ZMTX2/ZM2(1)
  18330=        COMMON/NDIMD/NNO,NRO,NPD,NMD.NDD,NWD.NWDD.NPLD.NWPNWD.NNPR
  18340=        COMMON/LOCD/LAP,LOP,LPHI,LBD,LEX,LPHD,LQ,LQN,LQD,LC,LDY,LEY
  18345=       1,LHP,LR
  18350=        COMMON/DSNMTX/NVDM,NODY,NOEY.DM(1)
  18360=        COMMON/LKF/LEADSN,LFLTRK,LFCOV
  18370=        COMMON/CKF/NVFLT.FLT(1)
  18380=        IF(NWPNWD.GT.0) GO TO 1
  18390=        WRITE(KTERM,108)
  18400= 108    FORMAT('ONO DRIVING NOISES - - FILTER DESIGN ABORT')
  18410=        RETURN
  18420= 1      IF(NMD.GT.0) GO TO 2
  18430=        WRITE(KTERM,109)
  18440= 109    FORMAT('ONO MEASUREMENTS - - FILTER DESIGN ABORT')
  18450=        RETURN
  18460= 2      WRITE(KLIST,110)
  18470= 110    FORMAT(////11X,5('* '),'FILTER DESIGN',5(' *')////)
  18480=        NSIZE=NPLD*(1+NPLD+NMD)
  18490=        IF(NSIZE.LE.NVFLT) GO TO 3
  18500=        WRITE 101,NSIZE
  18510= 101    FORMAT('OINSUFFICIENT MEMORY /CKF/, NEED: ',I4)
  18520=        LABORT=NSIZE
  18530=        RETURN
  18540= 3      NDIM=NPLD
  18550=        NDIM1=NDIM+1
  18560= 5      IF(NWD.EQ.0) GO TO 12
  18570=        IF(IFLTR.LE.0) GO TO 6
  18580=        WRITE 105,NWD
  18590= 105    FORMAT(' ENTER STATE NOISE STRENGTHS: ',I2)
  18600=        CALL RQWGTS(DM(LQ),NWD,0)
  18610= 6      CALL DVCTOR(NWD,DM(LQ),ZM1)
  18620=        CALL MATLST(ZM1,NWD,1,'Q',KTERM)
  18630= 10     CALL MATLST(DM(LQ),NWD,NWD,'Q',KLIST)
  18640= 12     IF(NWDD.EQ.0) GO TO 18
  18650=        IF(IFLTR.LE.0) GO TO 13
  18660=        WRITE 106,NWDD
  18670= 106    FORMAT(' ENTER DISTURBANCE NOISE STRENGTHS: ',I2)
  18680=        CALL RQWGTS(DM(LQN),NWDD,0)
  18690= 13     CALL DVCTOR(NWDD.DM(LQN).ZM1)
  18700=        CALL MATLST(ZM1,NWDD,1.'QN',KTERM)
  18710= 15     CALL MATLST(DM(LQN),NWDD,NWDD,'QN',KLIST)
  18720= 18     CALL QDSCRT(DM(LQ),DM(LQN),ZM1,ZM2)
A 18730=        CALL DAS(DM(LBD),DM(LQD),ZM1,ZM2)
  18740=        IF(IFLTR.LE.0) GO TO 19
  18750=        WRITE 107,NMD
  18760= 107    FORMAT(' ENTER MEASUREMENT NOISE STRENGTHS: ',I2)
```

```
      CALL ROWGTS(DM(LR),NMD,0)
 19   CALL DVCTOR(NMD,DM(LR),ZM1)
      CALL MATLST(ZM1,NMD,1,'R',KTERM)
 20   CALL MATLST(DM(LR),NMD,NMD,'R',KLIST)
 25   CALL TFRMTX(DM(LHP).SM,NMD,NDIM,2)
      CALL TRANS2(NMD,NDIM,SM,ZM1)
      LFCOV=LFLTRK+NDIM*NMD
      CALL DVCTOR(NMD,DM(LR),FLT(LFCOV))
      CALL KFLTR(NDIM,NMD,FLT,ZM1,DM(LOD),FLT(LFCOV),ZM2,
     1 FLT(LFLTRK),SM)
      CALL TFRMTX(SM,COM2,NDIM,NDIM.2)
      IA=1
      DO 30 I=1,NPLD
      FLT(LFCOV-1+I)=SQRT(ZM2(IA))
 30   IA=IA+NDIM1
      CALL MATLST(FLT(LFLTRK),NDIM,NMD,'KF',KLIST)
      CALL MATLST(FLT(LFLTRK),NDIM,NMD,'KF',KTERM)
      IFLTR=1
      LFLKF=1
 111  FORMAT(A3)
      RETURN
C END SUBROUTINE FLTRK
```

```
      SUBROUTINE PFDATA(ICODE,ND)
      COMMON/MAIN1/NDIM,NDIM1,COM1(1)
      COMMON/MAIN2/COM2(1)
      COMMON/INOU/KIN,KOUT,KPUNCH
      COMMON/DESIGN/NVCOM,TSAMP,LFLRPI,LFLCGT,LFLKF,LTEVAL,LABORT
      COMMON/FILES/KSAVE,KDATA,KPLOT,KLIST,KTERM
      COMMON/SYSMTX/NVSM,SM(1)
      COMMON/ZMTX1/NVZM,ZM1(1)
      COMMON/ZMTX2/ZM2(1)
      COMMON/NDIMD/NND,NRD,NPD.NMD.NDD,NWD.NWDD,NPLD,NWPNWD,NNPR
      COMMON/LOCD/LAP,LGP.LPHI,LRD,LEX,LPHD,LQ,LQN,LQD.LC,LDY,LEY
     1,LHP,LR
      COMMON/DSNMTX/NVDM.NODY.NOEY.DM(1)
      COMMON/NDIMC/NNC,NRC,NPC
      COMMON/LOCC/LPHC.LBDC,LCC.LDC
      COMMON/CMDMTX/NVCM,NEWCM,NODC,CM(1)
      COMMON/NDIMT/NNT,NRT,NMT,NWT
      COMMON/LOCT/LPHT,LBDT,LQDT,LHT,LRT,LTDT,LTNT
      COMMON/TRUMTX/NVTM,TM(1)
      COMMON/LCNTRL/LPI11,LPI12,LPI21,LPI22,LPHDL.LBDL
      COMMON/CONTROL/NVCTL,CTL(1)
      COMMON/LREGPI/LXDW,LUDW,LPHCL,LKX,LKZ
      COMMON/CREGPI/NVRPI,RPI(1)
      COMMON/LCGT/LA11,LA13,LA21,LA23,LA12,LA22,LKXA11.LKXA12,
     1LKXA13
      COMMON/CCGT/NVCGT,CGT(1)
      COMMON/LKF/LEADSN,LFLTRK,LFCOV
      COMMON/CKF/NVFLT,FLT(1)
      DIMENSION ND(1)
      DATA NO/1HN/
      WRITE 101
  101 FORMAT('OREAD IN CONTROLLER GAINS? (Y OR N) >')
      READ 103,IANS
  103 FORMAT(A3)
      IF (IANS.EQ.NO) GO TO 102
      CALL READFS(SM,ND,4,IERR)
      LKX=ND(3)
      LKZ=ND(4)
      LKXA11=ND(6)
      LKXA12=ND(7)
      LKXA13=ND(8)
      NNDP=(LKZ-LKX+1)/NRD
      CALL FTMTX(SM,RPI(LKX),NRD,NNDP)
      LL=LKX+NRD*NNDP
      LE=NND-NNDP
      CALL ZPART(RPI(LL),NRD,LE,NRD)
      LKZ=LL+NRD*LE
      LL=LL-LKX+1
      CALL FTMTX(SM(LL),RPI(LKZ),NRD.NPD)
      LL=ND(1)-ND(5)+1
      CALL FTMTX(SM(ND(5)),CGT(LKXA11),LL,1)
  102 ND(1)=NND
      ND(2)=NRD
      ND(3)=NPD
      ND(4)=NMD
      ND(5)=NDD
      ND(6)=NPLD
      ND(7)=NNC
      ND(8)=NRC
      ND(9)=NPC
      ND(10)=NNT
      ND(11)=NRT
      ND(12)=NMT
      ND(13)=NODY
      ND(14)=NOEY
      NVZMS=NVZM
C
      CALL FTMTX(DM(LPHI),SM,NND,NND)
      LL=NND*NND+1
      CALL FTMTX(DM(LBD),SM(LL),NND.NRD)
      LL=NND*NRD+LL
C
      IF(NDD.EQ.0) GO TO 100
           CALL FTMTX(DM(LEX),SM(LL),NND,NDD)
           LL=NND*NDD+LL
           CALL FTMTX(DM(LPHD),SM(LL),NDD,NDD)
           LL=NDD*NRD+LL
           IF(NODY.EQ.1) GO TO 90
                CALL FTMTX(DM(LDY),SM(LL),NPD,NRD)
                LL=NPD*NRD+LL
 90        IF(NOEY.EQ.1) GO TO 95
                CALL FTMTX(DM(LEY),SM(LL),NPD,NDD)
                LL=NPD*NDD+LL
 95        CALL FTMTX(DM(LHP),SM(LL),NMD,NPLD)
           LL=NMD*NPLD+LL
           GO TO 200
 100  CONTINUE
C
      IF(NODY.EQ.1) GO TO 105
           CALL FTMTX(DM(LDY),SM(LL),NPD,NRD)
           LL=NPD*NRD+LL
 105  CALL FTMTX(DM(LHP),SM(LL),NMD,NND)
      LL=NMD*NND+LL
 200  CALL FTMTX(DM(LC).SM(LL).NPD,NND)
      LL=NPD*NND+LL
      CALL FTMTX(CM(LPHC),SM(LL),NNC,NNC)
      LL=NNC*NNC+LL
      CALL FTMTX(CM(LBDC),SM(LL),NNC,NRC)
      LL=NNC*NRC+LL
      CALL FTMTX(CM(LCC),SM(LL),NPC,NNC)
      LL=NPC*NNC+LL
      CALL FTMTX(TM(LPHT),SM(LL).NNT,NNT)
      LL=NNT*NNT+LL
      CALL FTMTX(TM(LBDT),SM(LL),NNT,NRT)
      LL=NNT*NRT+LL
      CALL FTMTX(TM(LQDT),SM(LL),NNT,NNT)
      LL=NNT*NNT+LL
      CALL FTMTX(TM(LHT).SM(LL),NMT,NNT)
      LL=NMT*NNT+LL
      CALL FTMTX(TM(LRT),SM(LL),NMT,NMT)
      LL=NMT*NMT+LL
      CALL FTMTX(RPI(LKX),SM(LL),NRD,NND)
      LL=NRD*NND+LL
      CALL FTMTX(RPI(LKZ),SM(LL),NRD,NPD)
      LL=NRD*NPD+LL
      CALL FTMTX(CGT(LKXA11),SM(LL),NRC,NNC)
      LL=NRC*NNC+LL
C
      IF(NDD.EQ.0) GO TO 300
         CALL FTMTX(CGT(LKXA13),SM(LL),NRD,NDD)
         LL=NRD*NDD+LL
         CALL FTMTX(FLT(LFLTRK),SM(LL),NPLD,NMD)
         LL=NPLD*NMD+LL
         CALL FTMTX(TM(LTDT),SM(LL),NND,NNT)
         LL=NND*NNT+LL
         CALL FTMTX(TM(LTNT),SM(LL),NDD,NNT)
         LL=NDD*NNT+LL
         GO TO 310
 300  CONTINUE
C
      CALL FTMTX(FLT(LFLTRK),SM(LL),NND,NMD)
      LL=NND*NMD+LL
      CALL FTMTX(TM(LTDT),SM(LL),NND,NNT)
      LL=NND*NNT+LL
 310  SM(LL)=TSAMP
      ND(15)=LL
      CALL WFILED(ICODE,LL,ND,SM)
      NVZM=NVZMS
      RETURN
C END SUBROUTINE PFDATA
```

```
      SUBROUTINE DAS(BD,QD,BDP,QP)
      COMMON/MAIN1/NDIM
      COMMON/NDIMD/NND,NRD,NPD,NMD,NDD,NWD,NWDD,NPLD,NWFNWD,NNPR
      COMMON/FILES/KSAVE,KDATA,KPLOT,KLIST,KTERM
      DATA NO/1HN/
      WRITE 101
101   FORMAT("0MODIFY Q BY DOYLE AND STEIN? (Y OR N)> ")
      READ 102,IANS
102   FORMAT(A3)
      IF(IANS.EQ.NO) RETURN
      WRITE 103
103   FORMAT("0ENTER Q>")
      READ *,QDAS
      QDAS=QDAS*QDAS
      CALL TFRMTX(BD,BDP,NND,NRD,2)
      IF(NND.EQ.NPLD) GO TO 10
      L1=LADDR(NPLD,NRD+1,1)
10    CALL MAT2(NPLD,NRD,BDP,BDP,QP)
      CALL MADD1(NPLD,NPLD,QD,QP,QD,QDAS)
      CALL MATLST(QD,NPLD,NPLD,"QP",KLIST)
      RETURN
      END
```

A

APPENDIX D: Additional AFTI/F-16 Performance Data

D.1 Introduction

In Reference 6, it is stated that the Doyle and Stein technique for Kalman filter robustification is guaranteed to work only for a minimum-phase design model. That is, the model of the system to be controlled may not have transmission zeroes in the right-half s-plane.

The perturbation equations of motion described in Chapter V are linearized about a trim flight conditicn at an altitude of 10000 feet and a Mach number of 0.6. This yields a design model that is minimum phase. Initially, the trim condition was at the same altitude but with a Mach number of 0.8, to be consistent with other research done with the same model (Ref 12;27;29). At this design point, the eight-state controller model is non-minimum phase. The Doyle and Stein robustification was applied to this system with some unexpected results.

The results of a performance analysis of the eight-state continuous-time controller evaluated against a twelve-state truth model are presented in this chapter. First the perfcrmance was evaluated against a twelve-state truth model (10000 ft, M=0.6). Then, the response of the system at off-design condition is presented, with and without robustification.

D.2 Performance Evaluation of a Non-Minimum Phase Design Model

The performance of the non-minimum phase system at the design flight condition with a reduced-order design model is shown in Figure (D-1). The response of the system is stable with Θ converging to zero, and the standard deviation converging to a small finite value. The performance

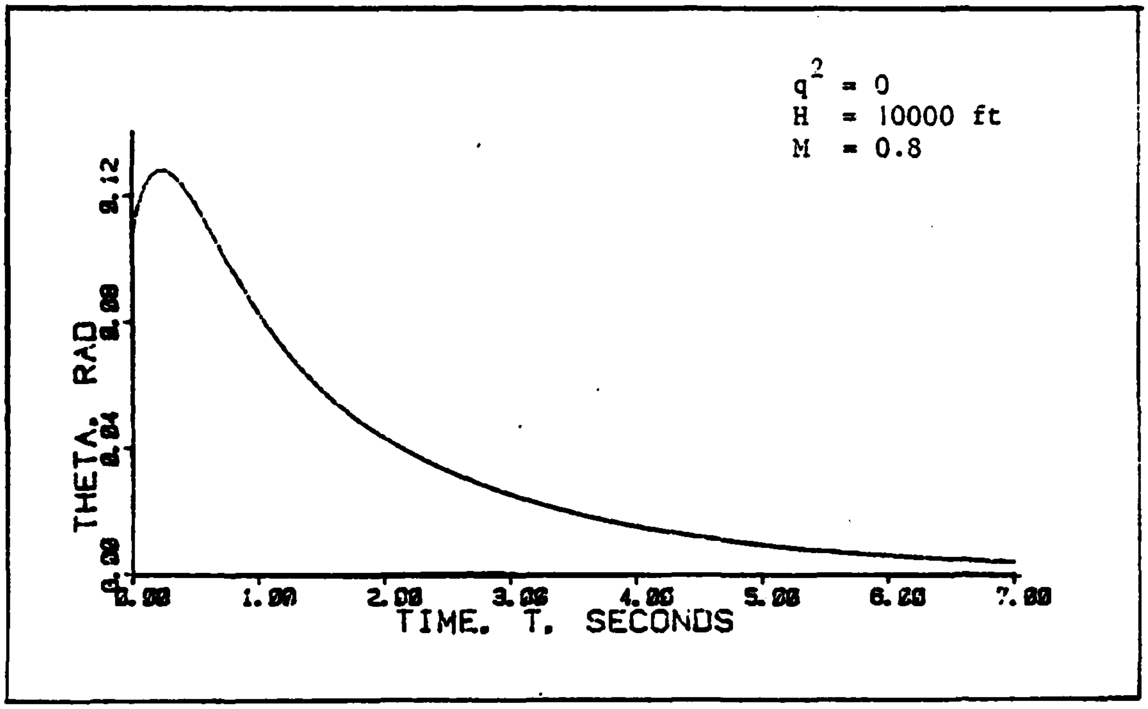


Figure D-1a: Unrobustified Non-Minimum Phase Mean Response of Θ

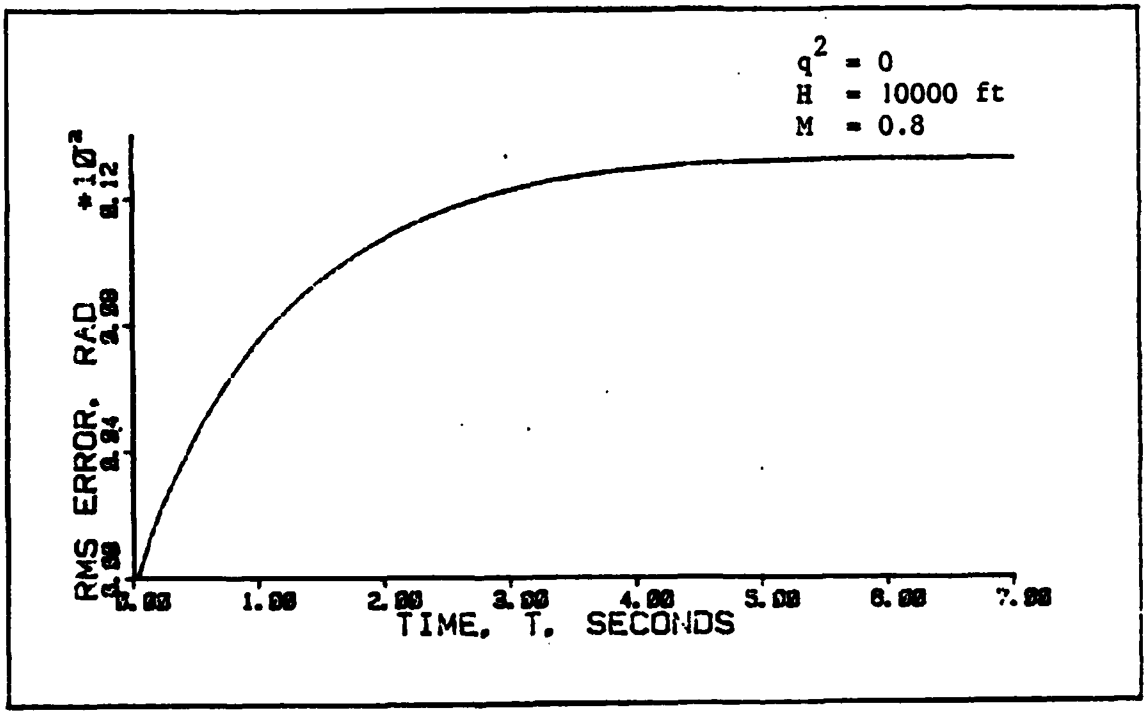


Figure D-1b: Standard Deviation of Theta for the Unrobustified Non-Minimum Phase System

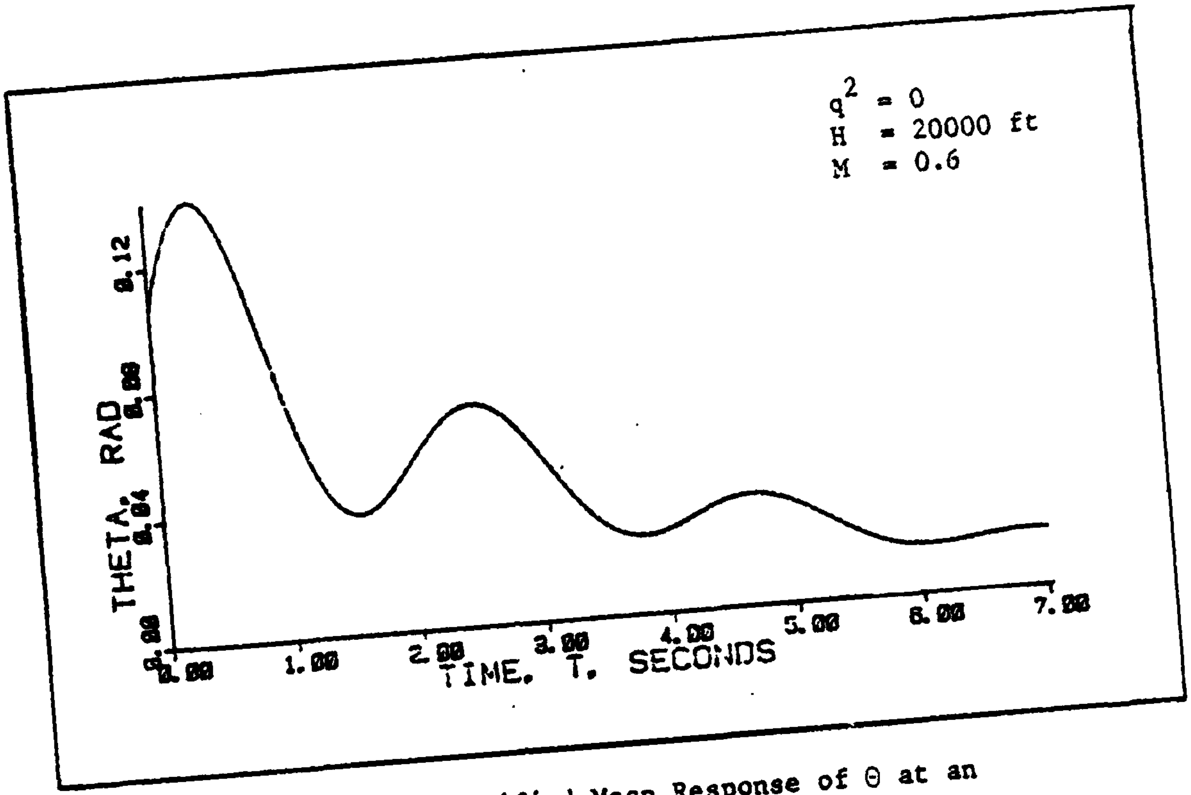


Figure D-2: Unrobustified Mean Response of Θ at an Off-Design Condition

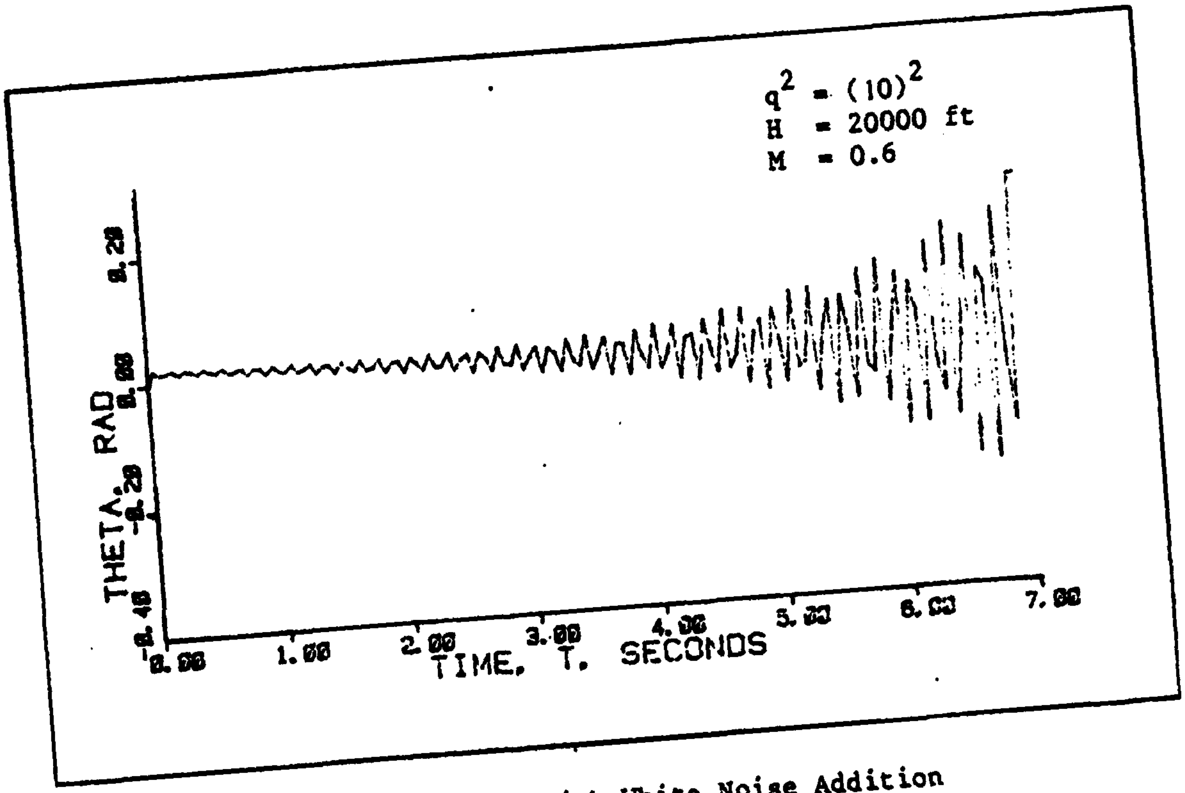


Figure D-3: Mean of Theta With White Noise Addition

of this controller is actually better than the one described in Chapters V and VI. The transient time and steady-state error are substantially better.

Figure (D-2) shows the mean of Θ response when the flight condition is changed to an altitude of 20000 feet and a Mach number of 0.6. This response is still stable although slower and more oscillatory than the previous case. Figures (D-3) through (D-5) show the response of the same controller with progressively higher strengths of white noise added to the model at the control entry points. It is seen that the stability of the controlled system is lost with the application of the robustification technique. Examination of the eigenvalues of the closed-loop system matrix disclosed that, for any non-zero value of q, some of the eigenvalues are driven into the right-half s-plane.

If the response of the system at another off-design point (10000 feet, M = 0.6) is examined (Figure D-6). It seen that the mean of Θ is diverging rapidly. However, as shown in Figure (D-7), if white noise of strength $q^2 = 1000$ is added to the system model, the divergence is considerably slower. Figures (D-8) and (D-9) demonstrate that adding white noise with higher strength will stabilize the pitch attitude response. The mean of Θ is actually converging to a steady-state value. Figures (D-10) through (D-12) show the same trend for the pitch rate, q.

The figures in this appendix show that the trends observed in Chapter VI do not apply if the model used for controller and Kalman filter design is non-minimum phase. A case was shown where the stability of the system at an off-design flight condition was recovered by applying

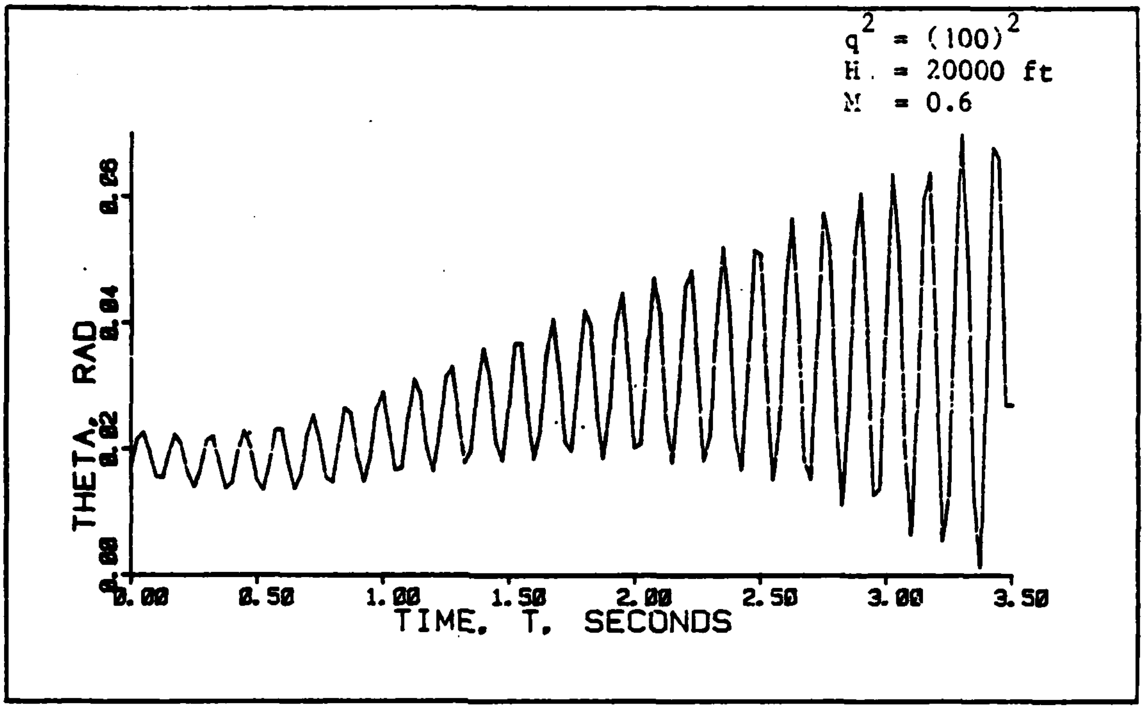


Figure D-4: Mean of Theta With White Noise Addition

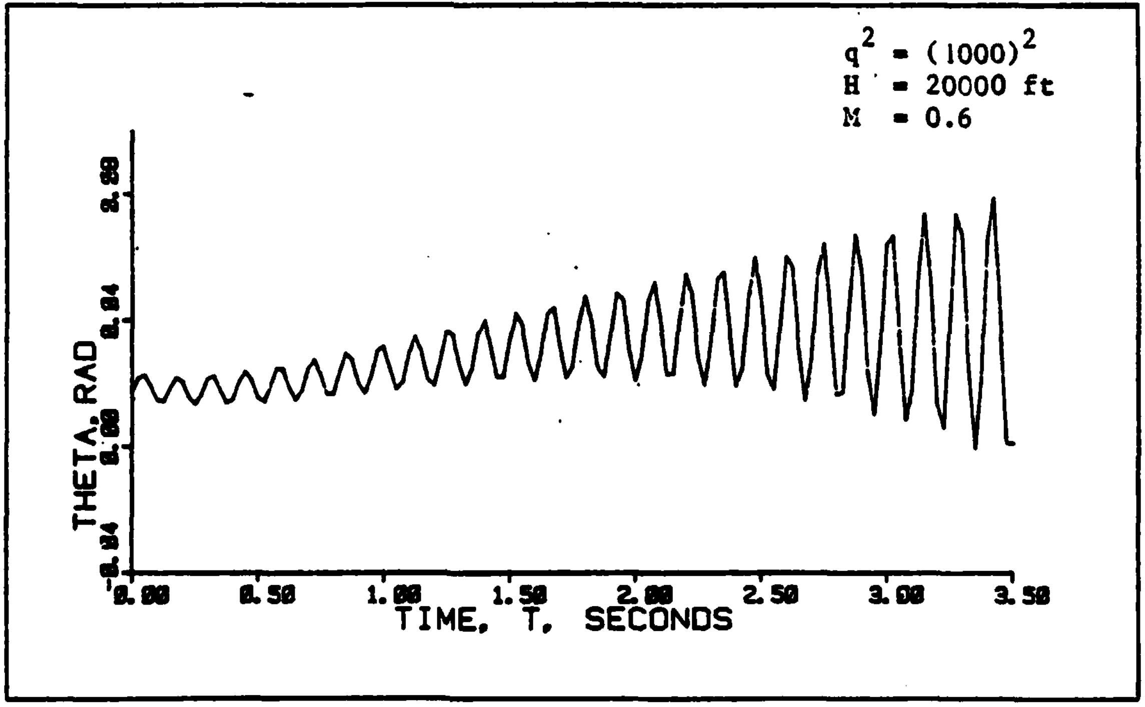


Figure D-5: Mean of Theta With White Noise Addition

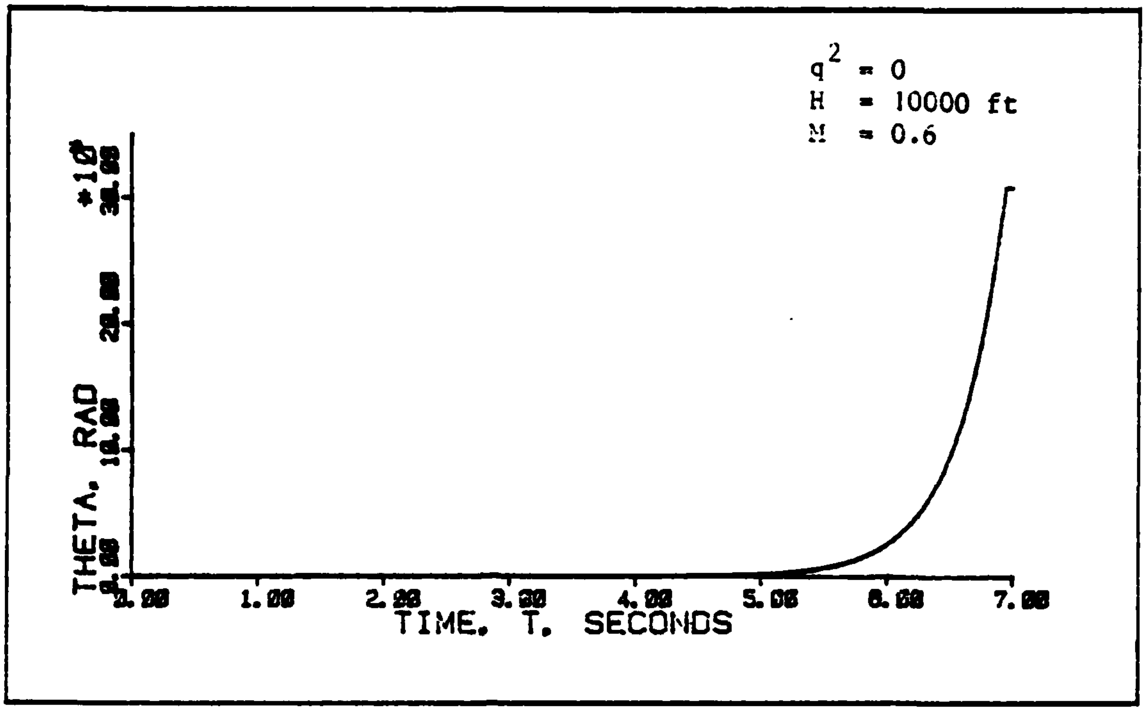


Figure D-6: Unrobustified Mean Response of Θ at an Off-Design Condition

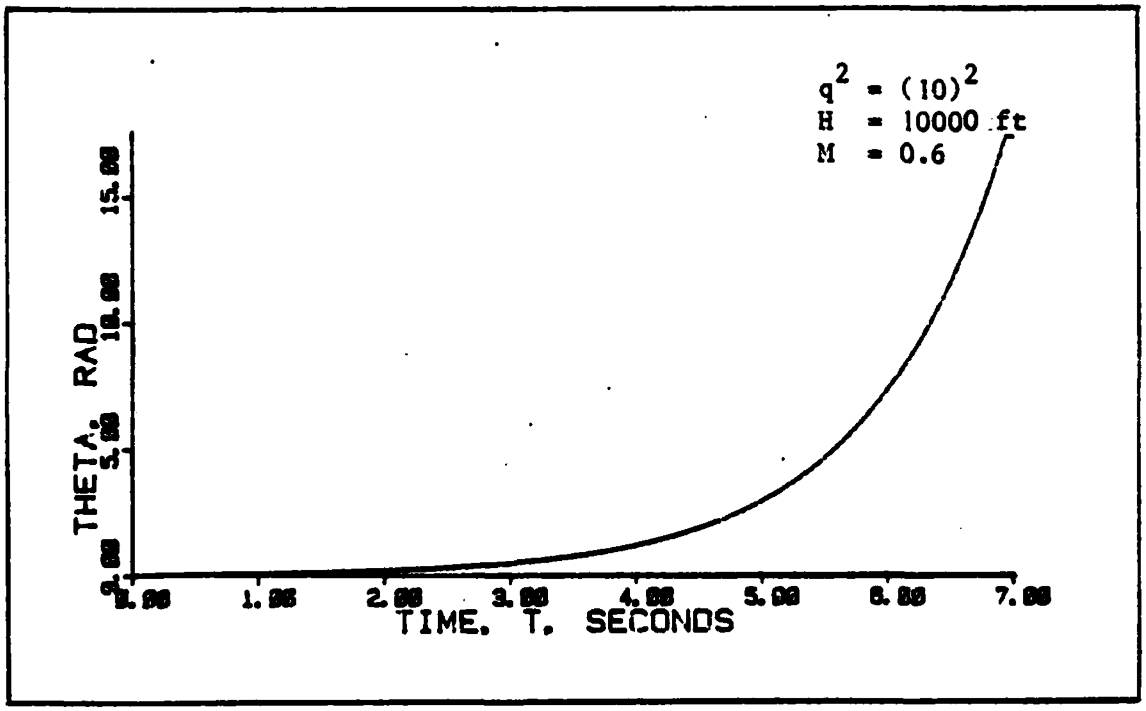


Figure D-7: Mean of Theta With White Noise Addition

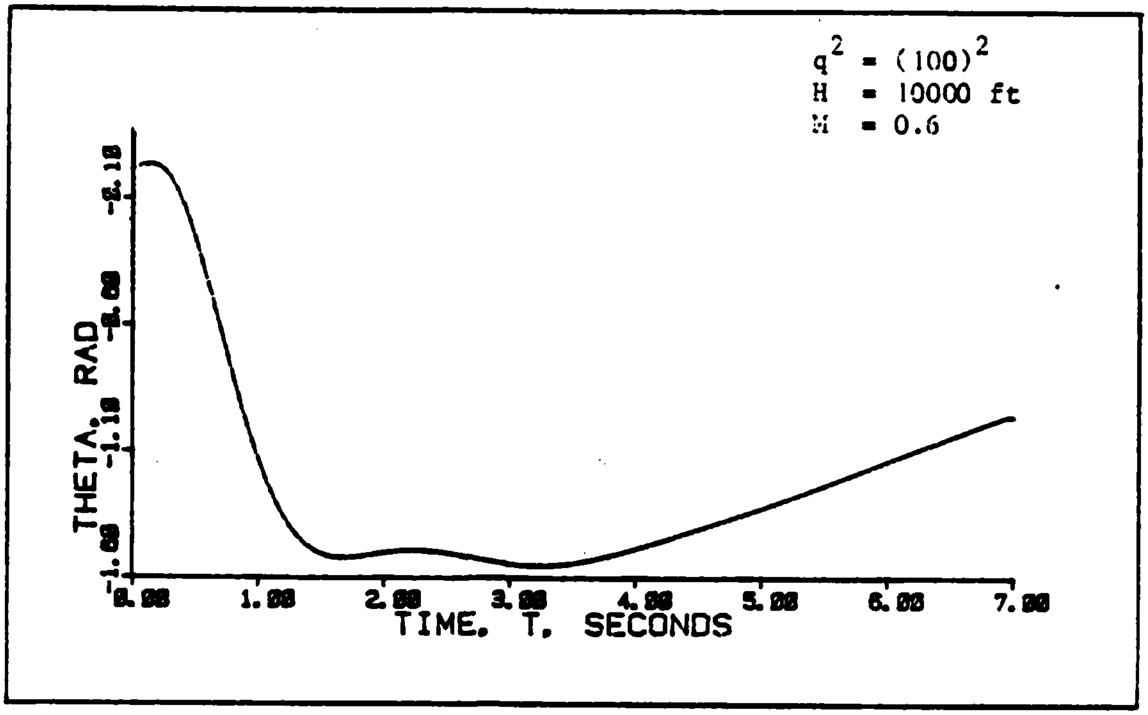


Figure D-8: Mean of Theta With White Noise Addition

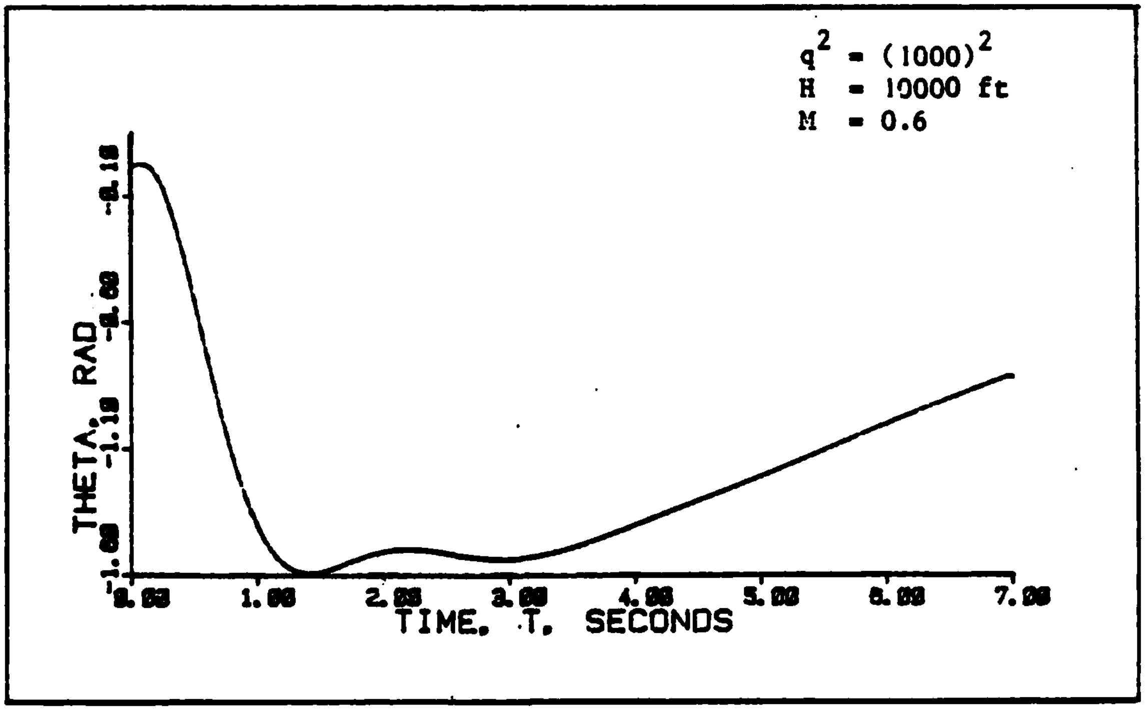


Figure D-9: Mean of Theta With White Noise Addition

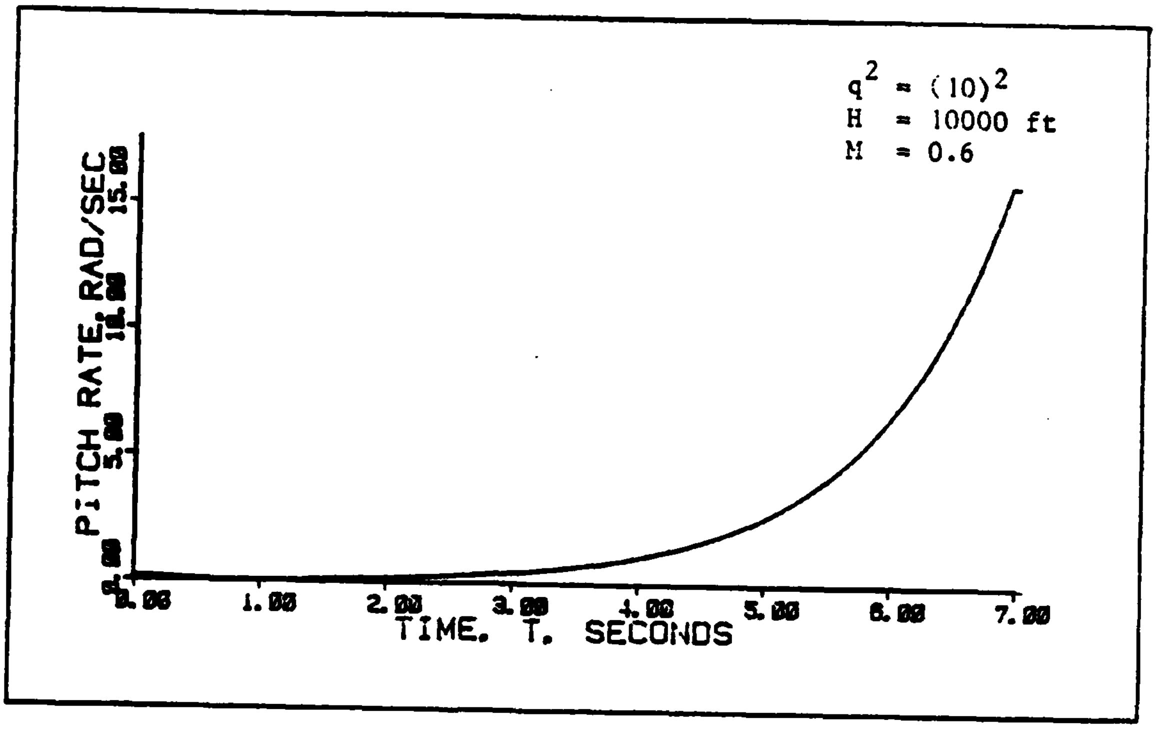


Figure D-10: Mean of q With White Noise Addition

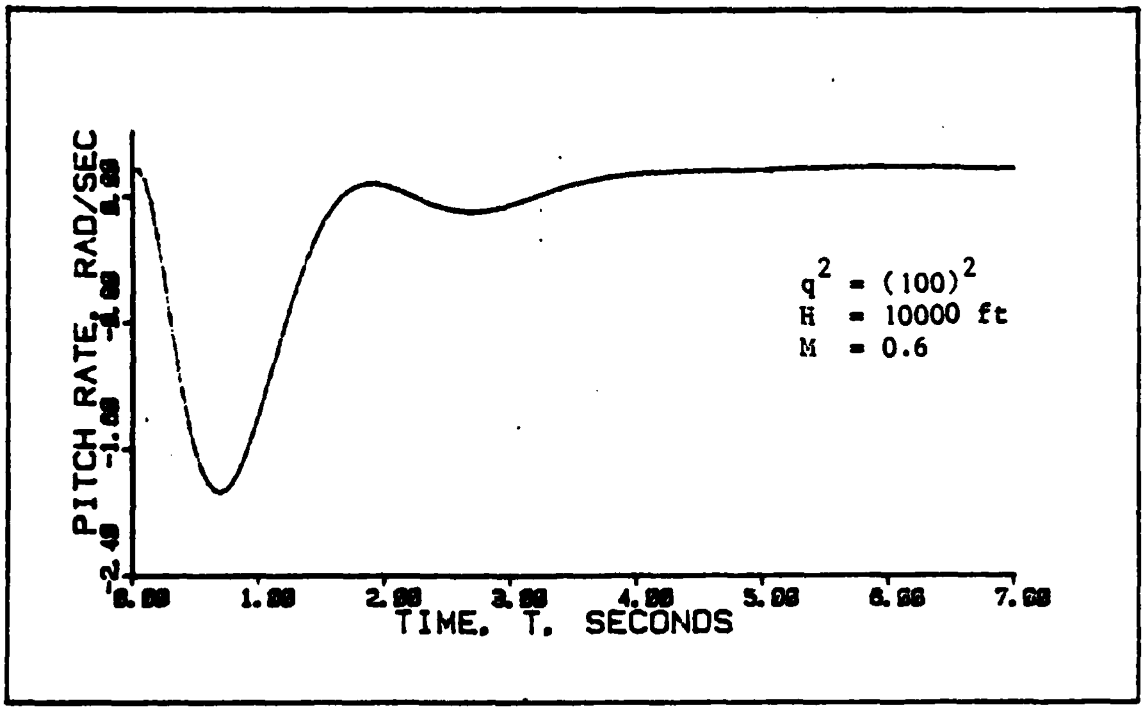


Figure D-11: Mean of q With White Noise Addition

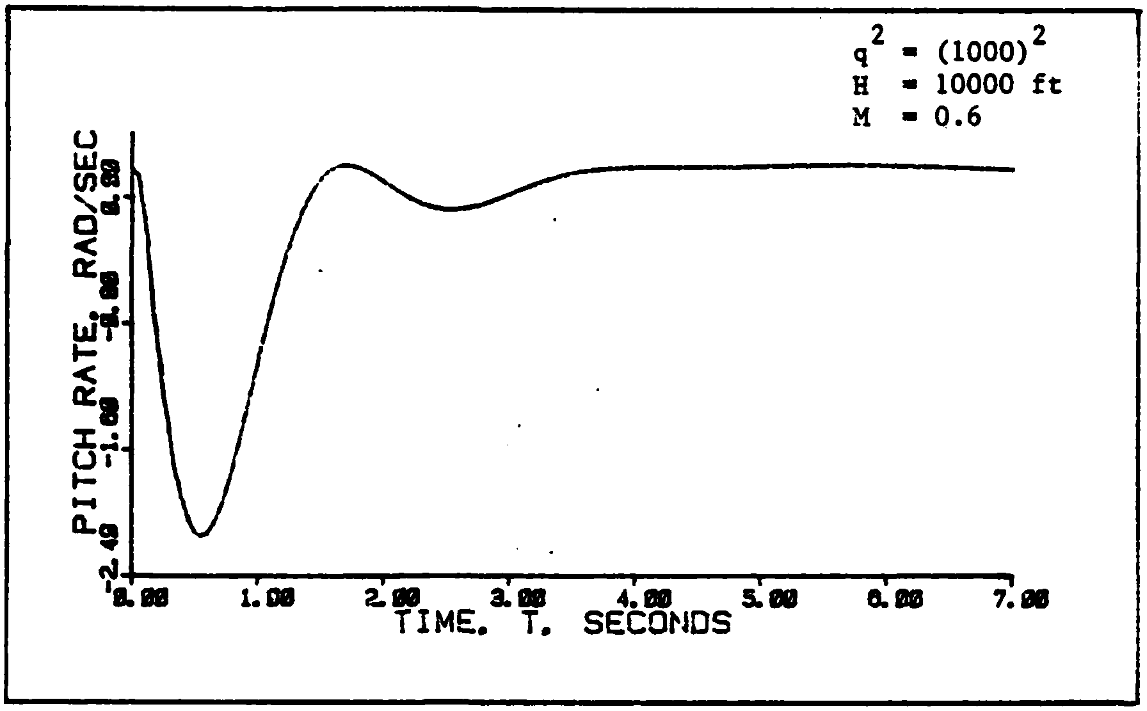


Figure D-12: Mean of q With White Noise Addition the robustification technique. However, the noise addition destabilized the response of the system at another flight condition that was initially stable.

Reference 28 deals with the robustification technique of adding time-correlated noise to a system model at the control entry points. It states that this method is not constrained to minimum phase models as in the Doyle and Stein technique. This claim was not examined in this thesis, however, the models used would form a good basis for future research in this area.

Vita

Jean Marie Howey was born on 16 June 1960 in Kansas City, Kansas. She graduated from high school in Meriden, Kansas in 1978. She attended the University of Kansas in Lawrence, Kansas and graduated with a Bachelor of Science in Aerospace Engineering in 1982. Upon graduation, she received a commission in the U.S. Air Force through the Reserve Officers Training Corps. She entered the School of Engineering of the Air Force Institute of Technology in June 1982 and has pursued the Stability and Control curriculum of the Aeronautical Engineering Department.

Permanent Address: 3221 Twilight Court, #103
Topeka, Kansas 66109

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION
UNCLASSIFIED

1b. RESTRICTIVE MARKINGS

2a. SECURITY CLASSIFICATION AUTHORITY

2b. DECLASSIFICATION/DOWNGRADING SCHEDULE

3. DISTRIBUTION/AVAILABILITY OF REPORT
Approved for public release;
distribution unlimited

4. PERFORMING ORGANIZATION REPORT NUMBER(S)
AFIT/GAE/EE/83D-2

5. MONITORING ORGANIZATION REPORT NUMBER(S)

6a. NAME OF PERFORMING ORGANIZATION
School of Engineering

6b. OFFICE SYMBOL *(If applicable)*
AFIT/EN

7a. NAME OF MONITORING ORGANIZATION

6c. ADDRESS *(City, State and ZIP Code)*
Air Force Institute of Technology
Wright-Patterson AFB, Ohio 45433

7b. ADDRESS *(City, State and ZIP Code)*

8a. NAME OF FUNDING/SPONSORING ORGANIZATION

8b. OFFICE SYMBOL *(If applicable)*

9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER

8c. ADDRESS *(City, State and ZIP Code)*

10. SOURCE OF FUNDING NOS.

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNI NO.

11. TITLE *(Include Security Classification)*
Robust Flight Controllers

12. PERSONAL AUTHOR(S)
Howey, Jean M., 2Lt, USAF

13a. TYPE OF REPORT
MS Thesis

13b. TIME COVERED
FROM ________ TO ________

14. DATE OF REPORT *(Yr., Mo., Day)*
1983 December

15. PAGE COUNT
270

16. SUPPLEMENTARY NOTATION

7 Feb 84

Approved for public release: IAW AFR 190-17
Lynn Wolaver
LYNN E. WOLAVER
Dean for Research and Professional Development
Air Force Institute of Technology (ATC)
Wright-Patterson AFB OH 45433

17. COSATI CODES

FIELD	GROUP	SUB. GR.

18. SUBJECT TERMS *(Continue on reverse if necessary and identify by block number)*
Robustness Enhancement, LQG Controller,
Kalman Filter, Optimal Control Theory

19. ABSTRACT *(Continue on reverse if necessary and identify by block number)*

This study examines the concept of robustifying a controlled system against differences which may exist between the real world system and the low-order design model upon which th controller design is based. The types of controllers considered are based upon the Linear system model, Quadratic cost, and Gaussian (LQG) noise process methodology of optimal control theory. It is assumed that full-state feedback is not available and a Kalman filt is employed to provide state estimates to the controller. Both continuous-time and sampl data controllers are considered.

Two robustification techniques are considered. The first is the method of injecting zer mean white Gaussian noise into the design model at the point of entry of the control input during the process of tuning the Kalman filter. The second method is an extension of the first, where the white noise is replaced by a time-correlated noise. This allows the primary strength of the noise to be concentrated only in the frequency range where robustific tion is desired. Comparing the results of applying the two methods allows a designer

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT
UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. □ DTIC USERS □

21. ABSTRACT SECURITY CLASSIFICATION
UNCLASSIFIED

22a. NAME OF RESPONSIBLE INDIVIDUAL
Peter S. Maybeck

22b. TELEPHONE NUMBER *(Include Area Code)*
513-255-3576

22c. OFFICE SYMBOL
AFIT/EN

EDITION OF 1 JAN 73 IS OBSOLETE.

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PA

Block #18

to make a trade-off between the amount of desired robustification and the performance degradation at the design conditions which occurs when the techniques are applied.

Both methods are found to improve substantially the robustness properties of the controllers considered. For the specific flight control problem considered in this thesis, the technique of injecting white input noise into the design model produced the desired degree of robustification without prohibitively degrading performance at the design conditions.

The second method, though effective, did not yield substantial enough performance benefits over the first to warrant use in actual implementation.

END

FILMED

3-84

DTIC